



Leseprobe

Holger Schwichtenberg

Windows PowerShell 5.0

Das Praxisbuch

ISBN (Buch): 978-3-446-44643-4

ISBN (E-Book): 978-3-446-44815-5

Weitere Informationen oder Bestellungen unter

<http://www.hanser-fachbuch.de/978-3-446-44643-4>

sowie im Buchhandel.

# Inhalt

<b>Vorwort zur fünften Auflage Januar 2016</b> .....	<b>XIX</b>
Wer bin ich? .....	XIX
Wer sind Sie? .....	XX
Was ist neu in diesem Buch? .....	XX
Sind in diesem Buch alle Features der PowerShell beschrieben? .....	XXI
Wem ist zu danken? .....	XXI
Woher bekommen Sie die Beispiele aus diesem Buch? .....	XXI
Wo kann ich mich schulen oder beraten lassen? .....	XXII
Zum Schluss des Vorworts ... ..	XXII
<b>Über den Autor Dr. Holger Schwichtenberg</b> .....	<b>XXIII</b>
<b>Über den Co-Autor Peter Monadjemi</b> .....	<b>XXIV</b>
<b>Teil A: PowerShell-Basiswissen</b> .....	<b>1</b>
<b>1 Erste Schritte mit der Windows PowerShell</b> .....	<b>3</b>
1.1 Was ist die Windows PowerShell? .....	3
1.2 Geschichte der Windows PowerShell .....	4
1.3 Motivation zur Windows PowerShell .....	5
1.4 Betriebssysteme mit vorinstallierter PowerShell .....	8
1.5 PowerShell und Linux .....	9
1.6 PowerShell herunterladen und auf anderen Betriebssystemen installieren ....	10
1.7 Die Windows PowerShell testen .....	13
1.8 Woher kommen die Commandlets? .....	22
1.9 PowerShell Community Extensions (PSCX) herunterladen und installieren ...	22
1.10 Den PowerShell-Editor „ISE“ verwenden .....	24
<b>2 Architektur der Windows PowerShell</b> .....	<b>29</b>

<b>3</b>	<b>Einzelbefehle der PowerShell</b> .....	<b>33</b>
3.1	Commandlets .....	33
3.2	Aliase .....	42
3.3	Ausdrücke .....	49
3.4	Externe Befehle .....	50
3.5	Dateinamen .....	51
<b>4</b>	<b>Hilfefunktionen</b> .....	<b>53</b>
4.1	Auflisten der verfügbaren Befehle .....	53
4.2	Erläuterungen zu den Befehlen .....	56
4.3	Hilfe zu Parametern .....	57
4.4	Hilfe mit Show-Command .....	57
4.5	Hilfefenster .....	58
4.6	Aktualisieren der Hilfedateien .....	60
4.7	Online-Hilfe .....	61
4.8	Dokumentation der .NET-Klassen .....	62
<b>5</b>	<b>Objektorientiertes Pipelining</b> .....	<b>65</b>
5.1	Pipeline-Operator .....	65
5.2	.NET-Objekte in der Pipeline .....	66
5.3	Pipeline Processor .....	68
5.4	Pipelining von Parametern .....	69
5.5	Pipelining von klassischen Befehlen .....	71
5.6	Anzahl der Objekte in der Pipeline .....	72
5.7	Zugriff auf einzelne Objekte aus einer Menge .....	72
5.8	Zugriff auf einzelne Werte in einem Objekt .....	74
5.9	Methoden ausführen .....	75
5.10	Analyse des Pipeline-Inhalts .....	77
5.11	Filtern .....	88
5.12	Zusammenfassung von Pipeline-Inhalten .....	91
5.13	„Kastrierung“ von Objekten in der Pipeline .....	92
5.14	Sortieren .....	93
5.15	Duplikate entfernen .....	93
5.16	Gruppierung .....	94
5.17	Berechnungen .....	96
5.18	Zwischenschritte in der Pipeline mit Variablen .....	96
5.19	Verzweigungen in der Pipeline .....	97
5.20	Vergleiche zwischen Objekten .....	99
5.21	Zusammenfassung .....	100
5.22	Praxisbeispiele .....	100

<b>6</b>	<b>PowerShell-Skripte</b> .....	<b>103</b>
6.1	Skriptdateien .....	103
6.2	Start eines Skripts .....	105
6.3	Aliase für Skripte verwenden .....	106
6.4	Parameter für Skripte .....	107
6.5	Skripte dauerhaft einbinden (Dot Sourcing) .....	108
6.6	Sicherheitsfunktionen für PowerShell-Skripte .....	109
6.7	Skripte anhalten .....	113
<b>7</b>	<b>PowerShell-Skriptsprache</b> .....	<b>115</b>
7.1	Hilfe zur PowerShell-Skriptsprache .....	115
7.2	Befehlstrennung .....	116
7.3	Kommentare .....	116
7.4	Variablen .....	117
7.5	Variablenbedingungen .....	124
7.6	Zahlen .....	125
7.7	Zeichenketten .....	127
7.8	Reguläre Ausdrücke .....	133
7.9	Datum und Uhrzeit .....	140
7.10	Arrays .....	141
7.11	Assoziative Arrays (Hash-Tabellen) .....	143
7.12	Operatoren .....	145
7.13	Überblick über die Kontrollkonstrukte .....	146
7.14	Schleifen .....	147
7.15	Bedingungen .....	152
7.16	Unterroutinen (Prozedur/Funktionen) .....	154
7.17	Eingebaute Funktionen .....	159
7.18	Fehlerbehandlung .....	160
7.19	Objektorientiertes Programmieren mit Klassen .....	167
<b>8</b>	<b>Ausgaben</b> .....	<b>171</b>
8.1	Ausgabe-Commandlets .....	171
8.2	Out-GridView .....	174
8.3	Standardausgabe .....	175
8.4	Einschränkung der Ausgabe .....	178
8.5	Seitenweise Ausgabe .....	178
8.6	Ausgabe einzelner Werte .....	179
8.7	Details zum Ausgabeoperator .....	181
8.8	Benutzerdefinierte Tabellenformatierung .....	184
8.9	Ausgabe von Methodenergebnissen und Unterobjekten in Pipelines .....	185

8.10	Ausgabe von Methodenergebnissen und Unterobjekten in Zeichenketten	186
8.11	Unterdrückung der Ausgabe	186
8.12	Ausgaben an Drucker	187
8.13	Ausgaben in Dateien	187
8.14	Umleitungen (Redirection)	188
8.15	Sprachausgabe	189
<b>9</b>	<b>Benutzereingaben</b>	<b>191</b>
9.1	Read-Host	191
9.2	Grafischer Eingabedialog	192
9.3	Dialogfenster	192
9.4	Authentifizierungsdialog	193
<b>10</b>	<b>Das PowerShell-Navigationsmodell</b>	<b>195</b>
10.1	Einführungsbeispiel: Navigation in der Registrierungsdatenbank	195
10.2	Provider und Laufwerke	196
10.3	Navigationsbefehle	198
10.4	Pfadangaben	199
10.5	Beispiel	201
10.6	Eigene Laufwerke definieren	202
<b>11</b>	<b>PowerShell-Werkzeuge</b>	<b>203</b>
11.1	PowerShell-Standardkonsole	203
11.2	PowerShell Integrated Scripting Environment (ISE)	210
11.3	PowerShell Script Analyzer	216
11.4	CIM Explorer for PowerShell ISE	222
11.5	PowerShell Web Access (PSWA)	223
11.6	ISE Steroids	229
11.7	PowerShellPlus	230
11.8	PoshConsole	233
11.9	PowerGUI	234
11.10	PowerShell Analyzer	235
11.11	PrimalScript	236
11.12	PowerShell Help	238
11.13	PowerShell Help Reader	238
11.14	PowerTab	239
11.15	NuGet Package Manager	239
11.16	PowerShell Tools for Visual Studio	240
11.17	PowerShell Remoting	241
11.18	Vergleich der Skripteditoren	241

<b>Teil B: PowerShell-Aufbauwissen</b> .....	<b>243</b>
<b>12 Fernausführung (Remoting)</b> .....	<b>245</b>
12.1 RPC-Fernabfrage ohne WS-Management .....	246
12.2 Anforderungen an PowerShell Remoting .....	247
12.3 Rechte für PowerShell-Remoting .....	248
12.4 Einrichten von PowerShell Remoting .....	249
12.5 Überblick über die Fernausführungs-Commandlets .....	251
12.6 Interaktive Fernverbindungen im Telnet-Stil .....	252
12.7 Fernausführung von Befehlen .....	253
12.8 Parameterübergabe an die Fernausführung .....	257
12.9 Fernausführung von Skripten .....	258
12.10 Ausführung auf mehreren Computern .....	259
12.11 Sitzungen .....	260
12.12 Implizites Remoting .....	264
12.13 Zugriff auf entfernte Computer außerhalb der eigenen Domäne .....	265
12.14 Verwaltung des WS-Management-Dienstes .....	268
12.15 PowerShell Direct für HyperV .....	270
12.16 Praxisbeispiel zu PowerShell Direct .....	272
<b>13 Verwendung von .NET-Klassen</b> .....	<b>275</b>
13.1 Microsoft Developer Network (MSDN) .....	275
13.2 Erzeugen von Instanzen .....	276
13.3 Parameterbehaftete Konstruktoren .....	278
13.4 Initialisierung von Objekten .....	279
13.5 Nutzung von Attributen und Methoden .....	280
13.6 Statische Mitglieder in .NET-Klassen und statische .NET-Klassen .....	282
13.7 Generische Klassen nutzen .....	284
13.8 Zugriff auf bestehende Objekte .....	285
13.9 Laden von Assemblies .....	286
13.10 Objektanalyse .....	287
13.11 Auflistungen (Enumerationen) .....	288
13.12 Verknüpfen von Aufzählungswerten .....	289
<b>14 Verwendung von COM-Klassen</b> .....	<b>291</b>
14.1 Erzeugen von Instanzen .....	291
14.2 Nutzung von Attributen und Methoden .....	292
14.3 Holen bestehender Instanzen .....	293
<b>15 Zugriff auf die Windows Management Instrumentation (WMI)</b> ...	<b>295</b>
15.1 Einführung in WMI .....	295

15.2	WMI in der Windows PowerShell .....	321
15.3	Abruf von WMI-Objektmengen .....	322
15.4	Fernzugriffe .....	323
15.5	Filtern und Abfragen .....	323
15.6	Liste aller WMI-Klassen .....	328
15.7	Hintergrundwissen: WMI-Klassenprojektion mit dem PowerShell-WMI-Objektadapter .....	328
15.8	Beschränkung der Ausgabeliste bei WMI-Objekten .....	332
15.9	Zugriff auf einzelne Mitglieder von WMI-Klassen .....	333
15.10	Werte setzen in WMI-Objekten .....	334
15.11	Umgang mit WMI-Datumsangaben .....	336
15.12	Methodenaufrufe .....	337
15.13	Neue WMI-Instanzen erzeugen .....	338
15.14	Instanzen entfernen .....	339
15.15	Commandlet Definition XML-Datei (CDXML) .....	340
<b>16</b>	<b>Dynamische Objekte .....</b>	<b>343</b>
16.1	Erweitern bestehender Objekte .....	343
16.2	Komplett dynamische Objekte .....	345
<b>17</b>	<b>Einbinden von C# und VB.NET .....</b>	<b>347</b>
<b>18</b>	<b>Win32-API-Aufrufe .....</b>	<b>349</b>
<b>19</b>	<b>Fehlersuche .....</b>	<b>353</b>
19.1	Detailinformationen .....	353
19.2	Einzelschrittmodus .....	354
19.3	Zeitmessung .....	355
19.4	Ablaufverfolgung .....	356
19.5	Kommandozeilenbasiertes Script-Debugging .....	356
<b>20</b>	<b>Transaktionen .....</b>	<b>359</b>
20.1	Commandlets für Transaktionen .....	359
20.2	Start und Ende einer Transaktion .....	360
20.3	Zurücksetzen der Transaktion .....	361
20.4	Mehrere Transaktionen .....	362
<b>21</b>	<b>Hintergrundaufträge („Jobs“) .....</b>	<b>363</b>
21.1	Voraussetzungen .....	363
21.2	Architektur .....	364
21.3	Starten eines Hintergrundauftrags .....	364

21.4	Hintergrundaufträge abfragen	365
21.5	Warten auf einen Hintergrundauftrag	366
21.6	Abbrechen und Löschen von Aufträgen	366
21.7	Analyse von Fehlermeldungen	367
21.8	Fernausführung von Hintergrundaufträgen	367
21.9	Praxisbeispiel	367
<b>22</b>	<b>Geplante Aufgaben und zeitgesteuerte Jobs</b>	<b>369</b>
22.1	Geplante Aufgaben (Scheduled Tasks)	369
22.2	Zeitgesteuerte Jobs	373
<b>23</b>	<b>PowerShell-Workflows</b>	<b>379</b>
23.1	Ein erstes Beispiel	379
23.2	Unterschiede zu einer Function bzw. einem Skript	384
23.3	Einschränkungen bei Workflows	384
23.4	Workflows in der Praxis	386
23.5	Workflows in Visual Studio erstellen	393
<b>24</b>	<b>Ereignissystem</b>	<b>411</b>
24.1	WMI-Ereignisse	411
24.2	WMI-Ereignisabfragen	411
24.3	WMI-Ereignisse seit PowerShell 1.0	413
24.4	Registrieren von WMI-Ereignisquellen seit PowerShell 2.0	414
24.5	Auslesen der Ereignisliste	415
24.6	Reagieren auf Ereignisse	417
24.7	WMI-Ereignisse ab PowerShell-Version 3.0	419
24.8	Registrieren von .NET-Ereignissen	419
24.9	Erzeugen von Ereignissen	420
<b>25</b>	<b>Datenbereiche und Datendateien</b>	<b>423</b>
25.1	Datenbereiche	423
25.2	Datendateien	425
25.3	Mehrsprachigkeit/Lokalisierung	426
<b>26</b>	<b>Desired State Configuration (DSC)</b>	<b>429</b>
26.1	Grundprinzipien	430
26.2	DSC für Linux	430
26.3	Ressourcen	431
26.4	Verfügbare DSC-Ressourcen	431
26.5	Eigenschaften einer Ressource	434
26.6	Aufbau eines DSC-Dokuments	434



26.7	Commandlets für die Arbeit mit DSC	435
26.8	Ein erstes DSC-Beispiel	435
26.9	Kompilieren und Anwendung eines DSC-Dokuments	436
26.10	Variablen in DSC-Dateien	438
26.11	Parameter für DSC-Dateien	439
26.12	Konfigurationsdaten	440
26.13	Entfernen einer DSC-Konfiguration	443
26.14	DSC Pull Server	446
26.15	DSC-Praxisbeispiel 1: IIS installieren	453
26.16	DSC-Praxisbeispiel 2: Software installieren	454
26.17	DSC-Praxisbeispiel 3: Software deinstallieren	457
26.18	Realisierung einer DSC-Ressource	457
26.19	Weitere Möglichkeiten	458
<b>27</b>	<b>PowerShell-Snap-Ins</b>	<b>459</b>
27.1	Einbinden von Snap-Ins	459
27.2	Liste der Commandlets	463
27.3	Doppeldeutige Namen	464
<b>28</b>	<b>PowerShell-Module</b>	<b>465</b>
28.1	Überblick über die Commandlets	465
28.2	Modularchitektur	466
28.3	Module aus dem Netz herunterladen und installieren mit PowerShellGet	467
28.4	Module manuell installieren	472
28.5	Auflisten der verfügbaren Module	473
28.6	Importieren von Modulen	474
28.7	Entfernen von Modulen	477
<b>29</b>	<b>Ausgewählte PowerShell-Erweiterungen</b>	<b>479</b>
29.1	PowerShell-Module in Windows 7 und Windows Server 2008 R2	479
29.2	PowerShell-Module in Windows 8.0 und Windows Server 2012	481
29.3	PowerShell-Module in Windows 8.1 und Windows Server 2012 R2	483
29.4	PowerShell-Module in Windows 10 und Windows Server 2016	485
29.5	Windows PowerShell Community Extensions (PSCX)	490
29.6	PowerShellPack	493
29.7	<i>www.IT-Visions.de</i> PowerShell Extensions	495
29.8	Quest Management Shell for Active Directory	496
29.9	Microsoft Exchange Server	497
29.10	System Center Virtual Machine Manager	498
29.11	PowerShell Management Library for Hyper-V (pshyperv)	499
29.12	Powershell Outlook Account Manager	500

29.13 PowerShell Configurator (PSConfig) .....	500
29.14 Weitere Erweiterungen .....	501
<b>30 Tipps und Tricks zur PowerShell .....</b>	<b>503</b>
30.1 Befehlsgeschichte .....	503
30.2 System- und Hostinformationen .....	504
30.3 Alle Anzeigen löschen .....	505
30.4 Anpassen der Eingabeaufforderung (Prompt) .....	505
30.5 Standardeinstellungen ändern mit Profilskripten .....	506
30.6 ISE erweitern .....	511
30.7 PowerShell für Gruppenrichtlinienskripte .....	512
30.8 Einblicke in die Interna der Pipeline-Verarbeitung .....	514
<b>Teil C: PowerShell im Praxiseinsatz .....</b>	<b>517</b>
<b>31 Dateisystem .....</b>	<b>519</b>
31.1 Laufwerke .....	520
31.2 Ordnerinhalte .....	525
31.3 Kurznamen .....	526
31.4 Dateisystemoperationen .....	527
31.5 Praxisbeispiel: Leere Ordner löschen .....	528
31.6 Papierkorb leeren .....	529
31.7 Dateieigenschaften lesen .....	529
31.8 Praxisbeispiel: Fotos nach Aufnahmedatum sortieren .....	530
31.9 Datei-Hash .....	531
31.10 Finden von Duplikaten .....	532
31.11 Dateieigenschaften verändern .....	534
31.12 Verknüpfungen im Dateisystem .....	536
31.13 Komprimierung .....	541
31.14 Dateisystemfreigaben .....	544
31.15 Überwachung des Dateisystems .....	555
31.16 Dateiversionsverlauf .....	556
31.17 Windows Server Backup .....	557
<b>32 Festplattenverschlüsselung mit BitLocker .....</b>	<b>559</b>
32.1 Übersicht über das BitLocker-Modul .....	560
32.2 Verschlüsseln eines Laufwerks .....	561
<b>33 Dokumente .....</b>	<b>563</b>
33.1 Textdateien .....	563
33.2 CSV-Dateien .....	564

33.3	Analysieren von Textdateien .....	567
33.4	INI-Dateien .....	570
33.5	XML-Dateien .....	570
33.6	HTML-Dateien .....	579
33.7	Binärdateien .....	579
<b>34</b>	<b>Datenbanken .....</b>	<b>581</b>
34.1	ADO.NET-Grundlagen .....	581
34.2	Beispieldatenbank .....	587
34.3	Datenzugriff mit den Bordmitteln der PowerShell .....	588
34.4	Datenzugriff mit den PowerShell-Erweiterungen .....	599
34.5	Datenbankzugriff mit SQLPS .....	602
34.6	Datenbankzugriff mit SQLPSX .....	602
<b>35</b>	<b>Microsoft-SQL-Server-Administration .....</b>	<b>603</b>
35.1	PowerShell-Integration im SQL Server Management Studio .....	604
35.2	SQL-Server-Laufwerk „SQLSERVER:“ .....	605
35.3	Die SQLPS-Commandlets .....	608
35.4	Die SQL Server Management Objects (SMO) .....	610
35.5	SQLPSX .....	613
35.6	Microsoft-SQL-Server-Administration mit der PowerShell in der Praxis .....	621
<b>36</b>	<b>ODBC-Datenquellen .....</b>	<b>627</b>
36.1	ODBC-Treiber und -Datenquellen auflisten .....	628
36.2	Anlegen einer ODBC-Datenquelle .....	629
36.3	Zugriff auf eine ODBC-Datenquelle .....	630
<b>37</b>	<b>Registrierungsdatenbank (Registry) .....</b>	<b>633</b>
37.1	Schlüssel auslesen .....	633
37.2	Schlüssel anlegen und löschen .....	634
37.3	Laufwerke definieren .....	634
37.4	Werte anlegen und löschen .....	635
37.5	Werte auslesen .....	636
37.6	Praxisbeispiel: Windows-Explorer-Einstellungen .....	636
37.7	Praxisbeispiel: Massenanlegen von Registry-Schlüsseln .....	637
<b>38</b>	<b>Computerverwaltung .....</b>	<b>639</b>
38.1	Computerinformationen .....	639
38.2	Computername und Domäne .....	641
38.3	Herunterfahren und Neustarten .....	642
38.4	Wiederherstellungspunkte verwalten .....	643

<b>39</b>	<b>Hardwareverwaltung</b>	<b>645</b>
39.1	Hardwarebausteine	645
39.2	Plug-and-Play-Geräte	647
39.3	Druckerverwaltung (ältere Betriebssysteme)	647
39.4	Druckerverwaltung (seit Windows 8 und Windows Server 2012)	648
<b>40</b>	<b>Softwareverwaltung</b>	<b>651</b>
40.1	Softwareinventarisierung	651
40.2	Installation von Anwendungen	654
40.3	Deinstallation von Anwendungen	655
40.4	Praxisbeispiel: Installationstest	655
40.5	Installationen mit PowerShell Package Management („OneGet“)	656
40.6	Versionsnummer ermitteln	659
40.7	Servermanager	660
40.8	Softwareeinschränkungen mit dem PowerShell-Modul „AppLocker“	671
<b>41</b>	<b>Prozessverwaltung</b>	<b>677</b>
41.1	Prozesse auflisten	677
41.2	Prozesse starten	678
41.3	Prozesse beenden	680
41.4	Warten auf das Beenden einer Anwendung	680
<b>42</b>	<b>Systemdienste</b>	<b>681</b>
42.1	Dienste auflisten	681
42.2	Dienstzustand ändern	683
42.3	Diensteigenschaften ändern	684
<b>43</b>	<b>Netzwerk</b>	<b>685</b>
43.1	Netzwerkkonfiguration (ältere Betriebssysteme)	685
43.2	Netzwerkkonfiguration (ab Windows 8 und Windows Server 2012)	687
43.3	DNS-Client-Konfiguration	690
43.4	DNS-Namensauflösung	693
43.5	Erreichbarkeit prüfen (Ping)	695
43.6	Windows Firewall	696
43.7	Remote Desktop (RDP) einrichten	702
43.8	E-Mails senden (SMTP)	703
43.9	Abruf von Daten von einem HTTP-Server	705
43.10	Aufrufe von SOAP-Webdiensten	706
43.11	Aufrufe von OData-Diensten	709
43.12	BITSTransfer	709

<b>44</b>	<b>Ereignisprotokolle</b> .....	<b>711</b>
14.1	Protokolleinträge auslesen .....	711
14.2	Ereignisprotokolle erzeugen .....	713
14.3	Protokolleinträge erzeugen .....	713
14.4	Protokollgröße festlegen .....	713
14.5	Protokolleinträge löschen .....	713
<b>45</b>	<b>Leistungsdaten</b> .....	<b>715</b>
45.1	Zugriff auf Leistungsindikatoren über WMI .....	715
45.2	Get-Counter .....	716
<b>46</b>	<b>Sicherheitseinstellungen</b> .....	<b>719</b>
46.1	Grundlagen .....	719
46.2	Zugriffsrechtelisten auslesen .....	724
46.3	Einzelne Rechteinträge auslesen .....	726
46.4	Besitzer auslesen .....	727
46.5	Benutzer und SID .....	727
46.6	Hinzufügen eines Rechteintrags zu einer Zugriffsrechteliste .....	731
46.7	Entfernen eines Rechteintrags aus einer Zugriffsrechteliste .....	733
46.8	Zugriffsrechteliste übertragen .....	735
46.9	Zugriffsrechteliste über SDDL setzen .....	735
<b>47</b>	<b>Active Directory</b> .....	<b>737</b>
47.1	Benutzer- und Gruppenverwaltung mit WMI .....	738
47.2	Einführung in System.DirectoryServices .....	739
47.3	Basiseigenschaften .....	750
47.4	Benutzer- und Gruppenverwaltung im Active Directory .....	752
47.5	Verwaltung der Organisationseinheiten .....	760
47.6	Suche im Active Directory .....	761
47.7	Navigation im Active Directory mit den PowerShell Extensions .....	768
47.8	Verwendung der Active-Directory-Erweiterungen von <i>www.IT-Visions.de</i> .....	769
47.9	PowerShell-Modul „Active Directory“ (ADPowerShell) .....	771
47.10	PowerShell-Modul „ADSDeployment“ .....	794
47.11	Informationen über die Active-Directory-Struktur .....	796
<b>48</b>	<b>Gruppenrichtlinien</b> .....	<b>799</b>
48.1	Verwaltung der Gruppenrichtlinien .....	800
48.2	Verknüpfung der Gruppenrichtlinien .....	801
48.3	Gruppenrichtlinienberichte .....	803
48.4	Gruppenrichtlinienvererbung .....	804
48.5	Weitere Möglichkeiten .....	805

<b>49</b>	<b>Virtuelle Systeme mit Hyper-V</b>	<b>807</b>
49.1	Das Hyper-V-Modul von Microsoft	808
49.2	Die ersten Schritte mit dem Hyper-V-Modul	810
49.3	Virtuelle Maschinen anlegen	814
49.4	Umgang mit virtuellen Festplatten	820
49.5	Konfiguration virtueller Maschinen	823
49.6	Dateien kopieren in virtuelle Systeme	827
49.7	PowerShell Management Library for Hyper-V (für ältere Betriebssysteme)	828
<b>50</b>	<b>Internet Information Server (IIS)</b>	<b>833</b>
50.1	Überblick	833
50.2	Navigationsprovider	835
50.3	Anlegen von Websites	837
50.4	Praxisbeispiel: Massenanlegen von Websites	838
50.5	Ändern von Eigenschaften von Websites	841
50.6	Anwendungspool anlegen	841
50.7	Virtuelle Verzeichnisse und IIS-Anwendungen	842
50.8	Website-Zustand ändern	843
50.9	Anwendungspools starten und stoppen	843
50.10	Löschen von Websites	844
<b>51</b>	<b>Microsoft Exchange Server</b>	<b>845</b>
51.1	Daten abrufen	845
51.2	Postfächer verwalten	846
51.3	Öffentliche Ordner verwalten	847
<b>52</b>	<b>Optimierungen und Problemlösungen</b>	<b>849</b>
52.1	PowerShell-Modul „TroubleshootingPack“	849
52.2	PowerShell-Modul „Best Practices“	853
<b>53</b>	<b>Grafische Benutzeroberflächen</b>	<b>855</b>
53.1	Eingabemasken	855
53.2	Universelle Objektdarstellung	857
53.3	Zwischenablage	859
53.4	WPF PowerShell Kit (WPK)	860
<b>Teil D: Profiwissen – Erweitern der PowerShell</b>		<b>871</b>
<b>54</b>	<b>Entwicklung von Commandlets in der PowerShell-Sprache</b>	<b>873</b>
54.1	Aufbau eines skriptbasierten Commandlets	873
54.2	Parameterfestlegung	876

54.3	Auszeichnung der Parameterdefinitionen .....	881
54.4	Dokumentation .....	884
<b>55</b>	<b>Entwicklung eigener Commandlets mit C# .....</b>	<b>887</b>
55.1	Technische Voraussetzungen .....	887
55.2	Grundkonzept der .NET-basierten Commandlets .....	889
55.3	Schrittweise Erstellung eines minimalen Commandlets .....	890
55.4	Erstellung eines Commandlets mit einem Rückgabeobjekt .....	898
55.5	Erstellung eines Commandlets mit mehreren Rückgabeobjekten .....	900
55.6	Erstellen eines Commandlets mit Parametern .....	904
55.7	Verarbeiten von Pipeline-Eingaben .....	906
55.8	Verkettung von Commandlets .....	909
55.9	Fehlersuche in Commandlets .....	913
55.10	Statusinformationen .....	916
55.11	Unterstützung für Sicherheitsabfragen .....	921
55.12	Festlegung der Hilfeinformationen .....	923
55.13	Erstellung von Commandlets für den Zugriff auf eine Geschäftsanwendung ...	927
55.14	Konventionen für Commandlets .....	928
55.15	Weitere Möglichkeiten .....	930
<b>56</b>	<b>Hosting der Windows PowerShell .....</b>	<b>931</b>
56.1	Voraussetzungen für das Hosting .....	932
56.2	Hosting mit PSHost .....	933
56.3	Vereinfachtes Hosting seit PowerShell 2.0 .....	936
<b>57</b>	<b>PowerShell-Module erstellen .....</b>	<b>939</b>
57.1	Erstellen eines Skriptmoduls .....	939
57.2	Erstellen eines Moduls mit Binärdateien .....	941
57.3	Erstellen eines Moduls mit Manifest .....	941
<b>Anhang</b>	<b>.....</b>	<b>949</b>
<b>A</b>	<b>Crashkurs „Objektorientierung“ .....</b>	<b>951</b>
<b>B</b>	<b>Crashkurs „.NET Framework“ .....</b>	<b>959</b>
<b>C</b>	<b>Literatur .....</b>	<b>969</b>
<b>D</b>	<b>Weitere Informationen im Internet .....</b>	<b>973</b>
<b>Stichwortverzeichnis</b>	<b>.....</b>	<b>975</b>

# Vorwort zur fünften Auflage

Liebe Leserin, lieber Leser,

willkommen zur fünften Auflage dieses PowerShell-Buchs! Das vor Ihnen liegende Buch behandelt die Windows-PowerShell-Version 5.0 von Microsoft sowie ergänzende Werkzeuge von Microsoft und Drittanbietern (z. B. PowerShell Community Extensions). Das Buch ist aber auch geeignet, wenn Sie noch PowerShell 2.0, 3.0 oder 4.0 einsetzen. Welche Funktionen neu hinzugekommen sind, wird jeweils erwähnt.

## ■ Wer bin ich?

Mein Name ist Holger Schwichtenberg, ich bin derzeit 43 Jahre alt und habe im Fachgebiet Wirtschaftsinformatik promoviert. Ich lebe (in Essen, im Herzen des Ruhrgebiets) davon, dass mein Team und ich im Rahmen unserer Firma [www.IT-Visions.de](http://www.IT-Visions.de) anderen Unternehmen bei der Entwicklung von .NET- und PowerShell-Anwendungen beratend und schulend zur Seite stehen. Zudem entwickeln wir im Rahmen der 5Minds IT-Solutions GmbH & Co. KG Software ([www.5Minds.de](http://www.5Minds.de)) im Auftrag von Unternehmen.

Es ist mein Hobby und „Nebenberuf“, IT-Fachbücher zu schreiben. Dieses Buch ist, unter Mitzählung aller nennenswerten Neuauflagen, das 63. Buch, das ich allein oder mit Co-Autoren geschrieben habe. Meine weiteren Hobbys sind Mountain Biking, Lauf-Sport, Fotografie und Reisen.

Natürlich verstehe ich das Bücherschreiben auch als Werbung für die Arbeit unserer Unternehmen und wir hoffen, dass der ein oder andere von Ihnen uns beauftragen wird, Sie durch Beratung, Schulung und Auftragsentwicklung zu unterstützen.



## ■ Wer sind Sie?

Damit Sie den optimalen Nutzen aus diesem Buch ziehen können, möchte ich – so genau es mir möglich ist – beschreiben, an wen sich dieses Buch richtet. Hierzu habe ich einen Fragebogen ausgearbeitet, mit dem Sie schnell erkennen können, ob das Buch für Sie geeignet ist.

Sind Sie Systemadministrator in einem Windows-Netzwerk?	<input type="radio"/> Ja <input type="radio"/> Nein
Laufen die für Sie relevanten Computer mit den von PowerShell 3.0, 4.0 oder 5.0 unterstützten Betriebssystemen? (Windows 7/8/8.1/10, Windows Server 2008/2008 R2/2012/2012 R2/2016)	<input type="radio"/> Ja <input type="radio"/> Nein
Sie besitzen zumindest rudimentäre Grundkenntnisse im Bereich des (objekt-orientierten) Programmierens?	<input type="radio"/> Ja <input type="radio"/> Nein
Wünschen Sie einen kompakten Überblick über die Architektur, Konzepte und Anwendungsfälle der PowerShell?	<input type="radio"/> Ja <input type="radio"/> Nein
Sie können auf Schritt-für-Schritt-Anleitungen verzichten?	<input type="radio"/> Ja <input type="radio"/> Nein
Sie können auf formale Syntaxbeschreibungen verzichten und lernen lieber an aussagekräftigen Beispielen?	<input type="radio"/> Ja <input type="radio"/> Nein
Sie erwarten nicht, dass in diesem Buch <b>alle</b> Möglichkeiten der PowerShell detailliert beschrieben werden?	<input type="radio"/> Ja <input type="radio"/> Nein
Sind Sie, nachdem Sie ein Grundverständnis durch dieses Buch gewonnen haben, bereit, Detailfragen in der Dokumentation der PowerShell, von .NET und WMI nachzuschlagen, da das Buch auf 1000 Seiten nicht alle Details erläutern kann?	<input type="radio"/> Ja <input type="radio"/> Nein

Wenn Sie alle obigen Fragen mit „Ja“ beantwortet haben, ist das Buch richtig für Sie. In anderen Fällen sollten Sie sich erst mit einführender Literatur beschäftigen.

## ■ Was ist neu in diesem Buch?

Gegenüber der vorherigen Auflage zur PowerShell 4.0 wurde das Buch um die neuen Funktionen in PowerShell 5.0 erweitert und inhaltlich optimiert. Zudem wurden die bestehenden Inhalte des Buchs an einigen Stellen erweitert und ich habe Praxisbeispiele aus meinen Arbeiten für Kunden und an unserer eigenen IT-Infrastruktur aus den letzten Jahren hinzugefügt.

## ■ Sind in diesem Buch alle Features der PowerShell beschrieben?

Die PowerShell umfasst mittlerweile über 1400 Commandlets mit jeweils zahlreichen Optionen. Zudem gibt es unzählige Erweiterungen mit vielen hundert weiteren Commandlets. Zudem existieren zahlreiche Zusatzwerkzeuge. Es ist allein schon aufgrund der Vorgaben des Verlags für den Umfang des Buchs nicht möglich, alle Commandlets hier auch nur zu erwähnen. Zudem habe ich – obwohl ich selbst fast jede Woche mit der PowerShell in der Praxis arbeite – immer noch nicht alle Commandlets jemals eingesetzt. Ich beschreibe in diesem Buch, was ich selbst in der Praxis verwende.

## ■ Wem ist zu danken?

Folgenden Personen möchte ich meinen Dank für ihre Mitwirkung an diesem Buch aussprechen:

- meinem Kollegen und Freund Peter Monadjemi, der rund 100 Seiten mit Beispielen zu der Vor-Vor-Auflage dieses Buchs beigetragen hat (Themen: Workflows, Bitlocker, ODBC, Hyper-V, DNS-Client, Firewall und SQL-Server-Administration),
- Frau Bauer-Schiewek und Frau Hasselbach, die dieses Buchprojekt beim Carl Hanser Verlag koordinieren und vermarkten,
- Frau Petra Kienle, die meine Tippfehler gefunden und sprachliche Ungenauigkeiten eliminiert hat,
- meiner Frau und meinen Kindern dafür, dass sie mir das Umfeld geben, um neben meinem Hauptberuf an Büchern wie diesem zu arbeiten.

## ■ Woher bekommen Sie die Beispiele aus diesem Buch?

Unter <http://www.powershell-doktor.de/leser> biete ich ein **ehrenamtlich betriebenes** Webportal für Leser meiner Bücher an. In diesem Portal können Sie

- die Codebeispiele aus diesem Buch herunterladen,
- eine PowerShell-Kurzreferenz „Cheat Sheet“ (zwei DIN-A4-Seiten als Hilfe für die tägliche Arbeit) kostenlos herunterladen,
- Feedback zu diesem Buch geben (Bewertung abgeben und Fehler melden) und
- technische Fragen in einem Webforum stellen.

Alle registrierten Leser erhalten auch Einladungen zu kostenlosen Community-Veranstaltungen sowie Vergünstigungen bei unseren öffentlichen Seminaren zu .NET und zur PowerShell. Bei der Registrierung müssen Sie das Kennwort **The Man in the High Castle** angeben.

## ■ Wo kann ich mich schulen oder beraten lassen?

Unter der E-Mail-Adresse *buero@IT-Visions.de* stehen mein Team und ich für Anfragen bezüglich Schulung, Beratung und Entwicklungstätigkeiten zur Verfügung.

## ■ Zum Schluss des Vorworts ...

wünsche ich Ihnen viel Spaß und Erfolg mit der PowerShell 5.0!

*Dr. Holger Schwichtenberg*

*Essen, im Januar 2016*

# Über den Autor

## Dr. Holger Schwichtenberg



- Studienabschluss Diplom-Wirtschaftsinformatik an der Universität Essen
- Promotion an der Universität Essen im Gebiet komponenten-basierter Softwareentwicklung
- Seit 1996 selbstständig als unabhängiger Berater, Dozent, Softwarearchitekt und Fachjournalist
- Leiter des Berater- und Dozententeams bei *www.IT-Visions.de*



- Leitung der Softwareentwicklung im Bereich Microsoft/.NET bei der 5minds IT-Solutions GmbH & Co. KG (*www.5minds.de*)



- Über 60 Fachbücher beim Carl Hanser Verlag, bei Microsoft Press und Addison-Wesley sowie mehr als 850 Beiträge in Fachzeitschriften
- Gutachter in den Wettbewerbsverfahren der EU gegen Microsoft (2006–2009)
- Ständiger Mitarbeiter der Zeitschriften *iX* (seit 1999), *dotnetpro* (seit 2000) und *Windows Developer* (seit 2010) sowie beim Online-Portal *heise.de* (seit 2008)
- Regelmäßiger Sprecher auf nationalen und internationalen Fachkonferenzen (z. B. Microsoft TechEd, Microsoft Summit, Microsoft IT Forum, BASTA, BASTA-on-Tour, .NET Architecture Camp, Advanced Developers Conference, Developer Week, OOP, DOTNET Cologne, VS One, NRW.Conf, Net.Object Days, Windows Forum)
- Zertifikate und Auszeichnungen von Microsoft:
  - Microsoft Most Valuable Professional (MVP)
  - Microsoft Certified Solution Developer (MCSD)
- Thematische Schwerpunkte:
  - Microsoft .NET Framework, Visual Studio, C#, Visual Basic
  - .NET-Architektur/Auswahl von .NET-Technologien

- Einführung von .NET Framework und Visual Studio/Migration auf .NET
- Webanwendungsentwicklung mit IIS, ASP.NET und JavaScript/AJAX
- Enterprise .NET, verteilte Systeme/Webservices mit .NET
- Relationale Datenbanken, XML, Datenzugriffsstrategien
- Objektrelationales Mapping (ORM), insbesondere ADO.NET Entity Framework
- Windows PowerShell (WPS) und Windows Management Instrumentation (WMI)
- Ehrenamtliche Community-Tätigkeiten:
  - Vortragender für die International .NET Association (INETA)
  - Betrieb diverser Community-Websites: [www.dotnetframework.de](http://www.dotnetframework.de), [www.entwicklerlexikon.de](http://www.entwicklerlexikon.de), [www.windows-scripting.de](http://www.windows-scripting.de), [www.aspnetdev.de](http://www.aspnetdev.de) u. a.
- Firmenwebsites: <http://www.IT-Visions.de> und <http://www.5minds.de>
- Weblog: <http://www.dotnet-doktor.de>
- Kontakt: [buero@IT-Visions.de](mailto:buero@IT-Visions.de) sowie Telefon 02 01-64 95 90-0

## Über den Co-Autor Peter Monadjemi



Peter Monadjemi ist freiberuflicher Trainer und Autor mit den Schwerpunkten PowerShell, Windows-Server-Administration und Softwareentwicklung auf der Grundlage des .NET Frameworks. Er kennt die PowerShell bereits seit der Zeit, in der sie als Vorabversion noch unter dem Projektnamen „Monad“ lief – ein Codename, der für ihn eine Art „Wink des Schicksals“ war. Seitdem hat er viel Zeit mit der PowerShell verbracht und Hunderte von Skripten geschrieben. Er lebt in Esslingen am Neckar.

# 12

## Fernausführung (Remoting)

Eine der schmerzlich vermissten Funktionen in der ersten Windows PowerShell (Version 1.0) war die generelle Unterstützung für Fernzugriffe auf andere Systeme bzw. Fernverwaltung von entfernten Systemen. Mit den Bordmitteln der PowerShell 1.0 konnte man im Wesentlichen nur über WMI via Distributed COM (DCOM) Daten von anderen Systemen abrufen. Eine generelle Möglichkeit zur Fernausführung von Commandlets und Skripten gab es nicht. Hier hat Microsoft nun in der Version 2.0 deutlich nachgebessert.

Seit PowerShell 2.0 gibt es über das Protokoll WS-Management (kurz: WS-Man) die Fernausführungsmöglichkeit („PowerShell Remoting“) für einzelne Commandlets und ganze Skripte.

WS-Management ist ein Netzwerkprotokoll auf Basis von XML-Webservices unter Verwendung des Simple Object Access Protocol (SOAP). WS-Management dient dem Austausch von Verwaltungsinformationen zwischen (heterogenen) Computersystemen. WS-Management ist ein Standard der Desktop Management Task Force (DMTF), der im Jahr 2006 verabschiedet wurde.

WS-Management bietet eine enge Verbindung zum Web Based Enterprise Management (WBEM) alias Windows Management Instrumentation (WMI).

Microsoft bietet eine Implementierung von WS-Management unter dem Namen „Windows Remote Management (WinRM)“ seit Windows XP und Windows Server 2003.

In Vista und Windows Server 2008 ist die Version 1.1 der Implementierung enthalten. Windows 7 und Windows Server 2008 R2 enthalten Version 2.0. Für Windows XP, Windows Vista sowie Windows Server 2003 (inkl. R2) und Windows Server 2008 gibt es ein Add-On für WinRM v2 mit Namen „Windows Management Framework“. Seit Windows 8.x und Windows Server 2012 (inkl. R2) gibt es WinRM 3.0, das es auch als Add-On für Windows 7 und Windows Server 2008/2008 R2 gibt <http://www.microsoft.com/en-us/download/details.aspx?id=34595>.

Für einen Fernaufruf müssen sowohl der lokale (der Aufrufer, der Client) als auch der entfernte Computer (der Aufgerufene, der Server) Windows Remote Management (WinRM) ab Version 2.0 unterstützen. Außerdem muss die PowerShell (ab Version 2.0) auf beiden Systemen installiert sein. WinRM benutzt im Standard die Ports 5985 (HTTP) und 5986 (HTTPS). Die Authentifizierung erfolgt im Normalfall über Kerberos, alternativ sind auch Basisauthentifizierung, Digest und NTLM möglich.

Die Verbindung zwischen Client und Server kann permanent oder temporär sein.

Mit PowerShell Remoting ist nicht nur ein Fernaufruf eines Computers, sondern auch gleichzeitig mehrerer Computer möglich. So kann man z. B. ein Skript gleichzeitig auf mehreren entfernten Systemen starten.

## ■ 12.1 RPC-Fernabfrage ohne WS-Management

Einige Commandlets in der PowerShell besitzen eingebaute Fernabfragemöglichkeiten abseits von WS-Management. Diese Commandlets haben einen Parameter „-Computername“ und die Fernaufrufmöglichkeiten des Betriebssystems, die auf einem Remote Procedure Call (RPC) basieren.

Folgende Commandlets besitzen den Parameter „-Computername“ und arbeiten mit DCOM:

- Clear-EventLog
- Limit-EventLog
- Get-Counter
- New-EventLog
- Get-EventLog
- Remove-EventLog
- Get-HotFix
- Restart-Computer
- Get-Process
- Show-EventLog
- Get-Service
- Show-Service
- Get-WinEvent
- Stop-Computer
- Get-WmiObject (schon in PowerShell 1.0 vorhanden)
- Write-EventLog

Der Fernaufruf mit vorgenannten Commandlets funktioniert auch, wenn WS-Management nicht installiert und konfiguriert ist.



**TIPP:** Derartige Commandlets findet man mit:

```
Get-Command | where { $_.parameters.keys -contains "ComputerName" -and
  $_.parameters.keys -notcontains "Session" }
```

Der folgende Befehl ermittelt vom Computer „F170“ alle Dienste, die mit dem Buchstaben „i“ beginnen.

```
Get-Service -ComputerName F170 i*
```

Die Abfrage mehrerer Computer ist nur nacheinander durch Übergabe in der Pipeline möglich, da man bei diesen Commandlets bei Computername kein Array als solchen übergeben kann.

*Falsch:*

```
Get-Service -ComputerName F173, F170 i*
```

*Richtig:*

```
"F171", "F172", "F173" | % { Get-Service i* -ComputerName $_ } | ft Name, status, -machinename
```

Über das Attribut `MachineName` kann man jeweils sehen, welcher der abgefragten Computer das Ergebnis geliefert hat.

```
PS C:\Users\hs.FBI> "F171", "F172", "F172", "F173" | % { Get-Service i* -computername $_ } | ft name, status, machinename
```

Name	Status	MachineName
idsvc	Stopped	F171
IRKEXT	Stopped	F171
IPBusEnum	Stopped	F171
iphlpvc	Running	F171
idsvc	Stopped	F172
IRKEXT	Running	F172
IPBusEnum	Stopped	F172
iphlpvc	Running	F172
idsvc	Stopped	F172
IRKEXT	Running	F172
IPBusEnum	Stopped	F172
iphlpvc	Running	F172
idsvc	Stopped	F173
IRKEXT	Running	F173
IPBusEnum	Stopped	F173
iphlpvc	Running	F173

```
PS C:\Users\hs.FBI> _
```

**Bild 12.1** Abfrage der Dienste auf zwei Computern

## 12.2 Anforderungen an PowerShell Remoting

Für einen Fernaufruf via PowerShell Remoting müssen sowohl der lokale (der Aufrufer, der Client) als auch der entfernte Computer (der Aufgerufene, der Server) folgende Voraussetzungen erfüllen:

- Microsoft .NET Framework 2.0 oder höher
- Windows PowerShell 2.0 oder höher
- Windows Remote Management (WinRM) 2.0 oder höher



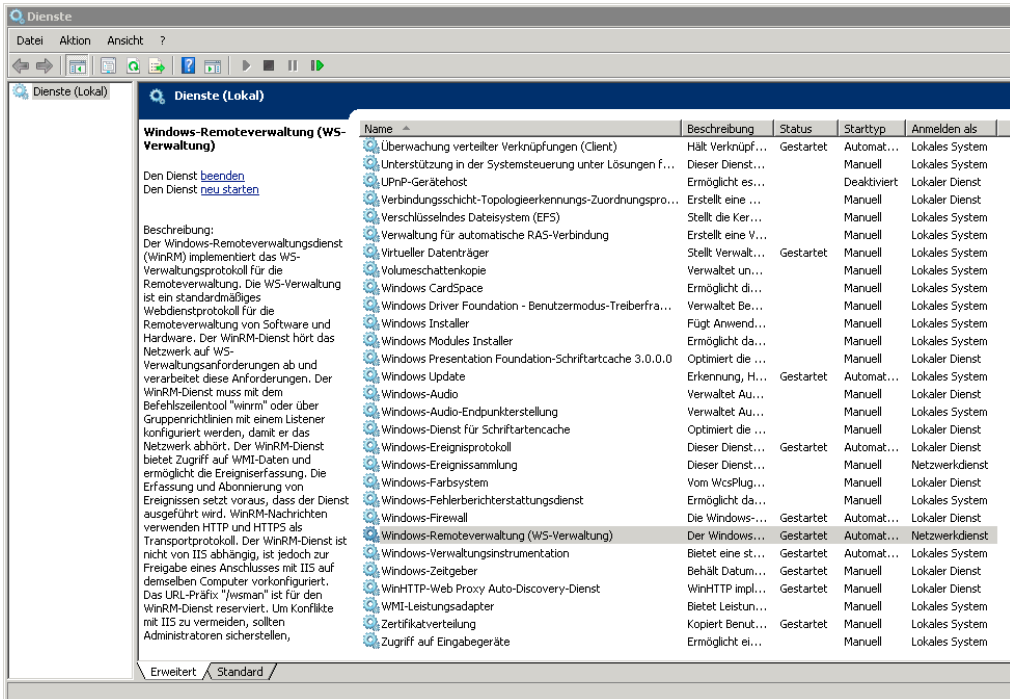


Bild 12.2 Der WinRM-Dienst in der Dienstliste

## 12.3 Rechte für PowerShell-Remoting

Fernaufrufe sind sowohl domänenintern als auch domänenübergreifend (durch Vertrauensstellungen oder gleiche Benutzername-Kennwort-Kombination) möglich, im zweiten Fall allerdings nur unter expliziter Angabe von Benutzernamen und Kennwort, selbst wenn auf dem Zielsystem eine zu dem aufrufenden System identische Benutzername-Kennwort-Kombination existiert. Fernaufrufe können nur Benutzer ausführen, die auf dem Zielsystem zur Administratorengruppe gehören. Man kann das Recht zum Fernaufwurf durch Änderung der sogenannten Sitzungskonfigurationen steuern.

Auch auf dem lokalen Computer werden Administratorrechte für einige Aktionen im Zusammenhang mit dem Fernaufwurf benötigt. Dies sind insbesondere die Einrichtung von WS-Management und die Konfiguration von PowerShell-Sitzungen. Auch ein Fernaufwurf gegen den eigenen Computer („Loopback-Aufruf“) erfordert Administratorrechte.

## ■ 12.4 Einrichten von PowerShell Remoting

Im Auslieferungszustand der PowerShell sind PowerShell-Fernaufrufe deaktiviert. Mit `Enable-PSRemoting` konfiguriert man einen Computer zum Empfang von Fernaufrufen von anderen Rechnern. Dieser Befehl startet den WinRM-Systemdienst (Listener), konfiguriert die Windows PowerShell und trägt das Protokoll WS-Management als Ausnahme in der Windows Firewall ein. `Enable-PSRemoting` ist nicht notwendig auf Computern, die nur (!) PowerShell-Befehle an andere Rechner senden wollen. Zum Ausführen von `Enable-PSRemoting` muss man auf dem System Administrator sein.

```
PS C:\Users\Administrator> Enable-PSRemoting

WinRM Quick Configuration
Running command "Set-WSManQuickConfig" to enable remote management of this computer by using the Windows Remote
Management (WinRM) service.
This includes:
  1. Starting or restarting (if already started) the WinRM service
  2. Setting the WinRM service startup type to Automatic
  3. Creating a listener to accept requests on any IP address
  4. Enabling Windows Firewall inbound rule exceptions for WS-Management traffic (for http only).

Do you want to continue?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
WinRM is already set up to receive requests on this computer.
WinRM is already set up for remote management on this computer.

Confirm
Are you sure you want to perform this action?
Performing the operation "Set-PSessionConfiguration" on target "Name: microsoft.powershell SDDL:
0-MSG:BAD:P(A;;GA;;;BA)(A;;GA;;;RM)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)". This lets selected users remotely run Windows
PowerShell commands on this computer."
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y

Confirm
Are you sure you want to perform this action?
Performing the operation "Set-PSessionConfiguration" on target "Name: microsoft.powershell.workflow SDDL:
0-MSG:BAD:P(A;;GA;;;BA)(A;;GA;;;RM)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)". This lets selected users remotely run Windows
PowerShell commands on this computer."
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y

Confirm
Are you sure you want to perform this action?
Performing the operation "Set-PSessionConfiguration" on target "Name: microsoft.powershell32 SDDL:
0-MSG:BAD:P(A;;GA;;;BA)(A;;GA;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)". This lets selected users remotely run Windows
PowerShell commands on this computer."
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y

Confirm
Are you sure you want to perform this action?
Performing the operation "Set-PSessionConfiguration" on target "Name: microsoft.windows.servermanagerworkflows SDDL:
0-MSG:BAD:P(A;;GA;;;BA)(A;;GA;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)". This lets selected users remotely run Windows
PowerShell commands on this computer."
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
PS C:\Users\Administrator>
```

Bild 12.3 Erstkonfiguration von PowerShell Remoting mit `Enable-PSRemoting`



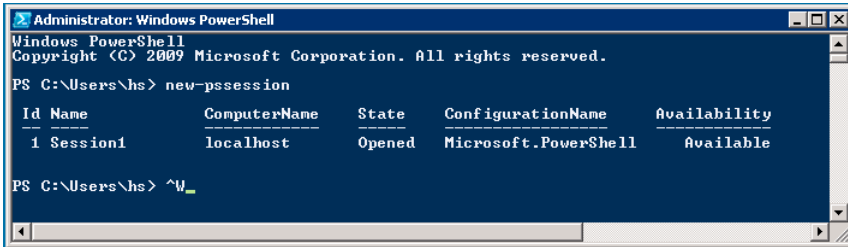
**TIPP:** Zum Unterdrücken der Nachfragen geben Sie ein:

`Enable-PSRemoting -force`

Zum Testen der Einrichtung geben Sie ein:

`New-PSSession`

Dann sollte das nachstehende Ergebnis erscheinen.



```

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\hs> new-pssession

Id Name          ComputerName State      ConfigurationName Availability
-- --          -
1 Session1     localhost  Opened    Microsoft.PowerShell Available

PS C:\Users\hs> ^W

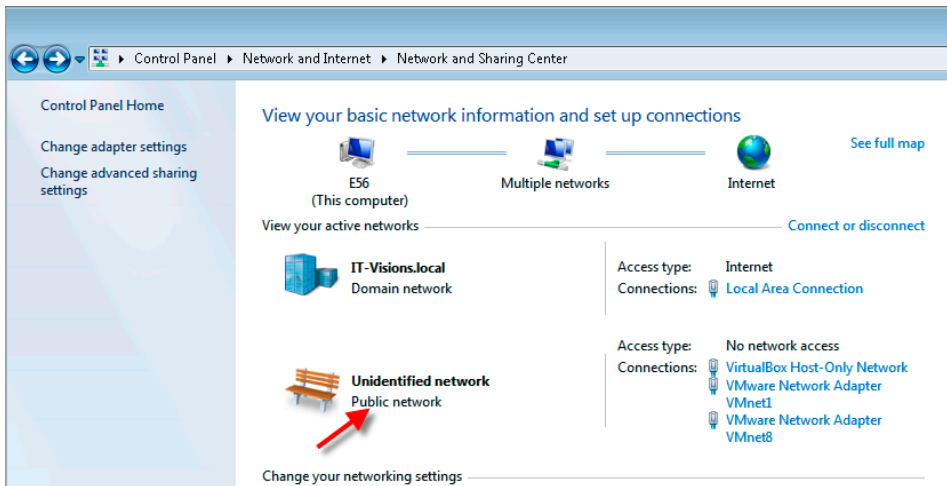
```

**Bild 12.4** Testen der Fernverbindungskonfiguration mit New-PSSession



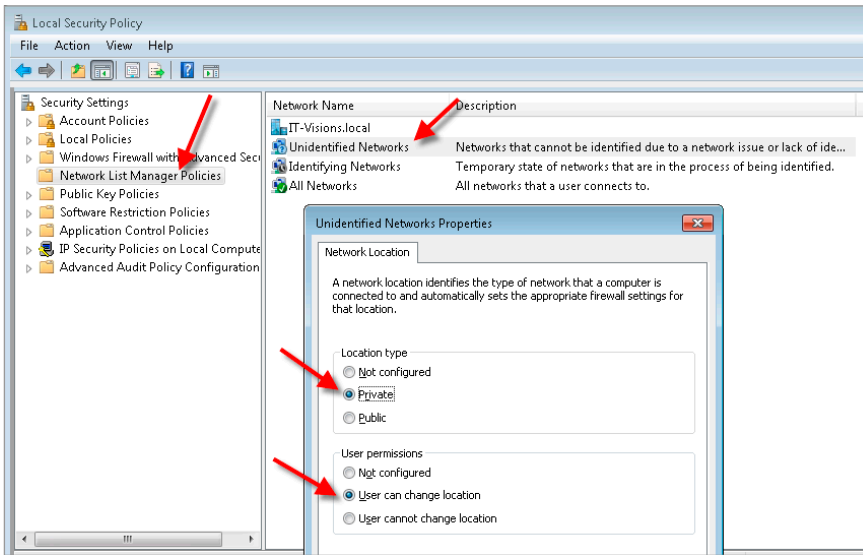
**HINWEIS:** In Domänen kann WinRM über die Gruppenrichtlinie „Computer Configuration\Administrative Templates\Windows Components\Windows Remote Management (WinRM)\WinRM service“ konfiguriert werden.

Ein Problem bei der Einrichtung könnte sein, dass diese mit der Fehlermeldung „Set-WSManQuickConfig : WinRM firewall exception will not work since one of the network connection types on this machine is set to Public. Change the network connection type to either Domain or Private and try again.“ abbricht. Ein Blick in das Netzwerkcenter von Windows wird dann zeigen, dass es tatsächlich ein „Public“-Netzwerk gibt (siehe Beispiel in der Abbildung).



**Bild 12.5** Ein „Public Network“ verhindert die Einrichtung des PowerShell Remoting.

Lösen kann man dies über secpol.msc (siehe Abbildung 12.6).



**Bild 12.6** Lösung des „Public Network“-Problems



**TIPP:** Seit PowerShell-Version 4.0 kann man alternativ beim `Enable-PSRemoting`-Commandlet auch den Parameter `SkipNetworkProfileCheck` angeben, wodurch die Prüfung auf das Netzwerkprofil entfällt.

Zum Deaktivieren des PowerShell Remoting gibt es das Commandlet:

```
Disable-PSRemoting
```

Durch `Disable-PSRemoting` wird der WinRM-Dienst aber nicht beendet.

## ■ 12.5 Überblick über die Fernausführungs-Commandlets

Die wichtigsten Commandlets für PowerShell Remoting sind:

- `Enter-PSSession`: Starten einer Fernausführungssitzung im Telnet-Stil
- `Invoke-Command`: Fernausführung eines einzelnen PowerShell-Commandlets oder eines Skripts
- `New-PSSession`: Erstellen einer permanenten Verbindung für die Fernausführung

Alle oben genannten Commandlets bieten ein Attribut `-Computername`. Bei `Enter-PSSession` kann man nur einen Computer angeben, bei `Invoke-Command` und `New-PSSession` auch ein Array mehrerer Computer.



**HINWEIS:** Im Standard vorgesehen ist die Angabe von Computernamen. IP-Adressen können alternativ verwendet werden, erfordern aber ein anderes Authentifizierungsverfahren, da das im Standard verwendete Kerberos keine IP-Adressen unterstützt (siehe Kapitel zur Sicherheit bei PowerShell Remoting).

## ■ 12.6 Interaktive Fernverbindungen im Telnet-Stil

Mit dem Commandlet `Enter-PSSession` eröffnet man eine interaktive Sitzung zu einem entfernten System im Stil des Telnet-Protokolls. Anzugeben ist der Computername, z. B.

```
Enter-PSSession -Computername F170
```

Nach erfolgreicher Ausführung des Befehls wird der Computername vor der PowerShell-Eingabeaufforderung angezeigt. Alle eingegebenen Befehle werden nun auf dem entfernten System ausgeführt. Alle Ausgaben landen auf dem lokalen System.

Testen kann man zum Beispiel, indem man mit `[System.Environment]::MachineName` den Computernamen abrufen.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\hs> [System.Environment]::MachineName
E03
PS C:\Users\hs> Enter-PSSession -Computer F170
[F170]: PS C:\Users\hs.ITU\Documents> [System.Environment]::MachineName
F170
[F170]: PS C:\Users\hs.ITU\Documents> _
```

**Bild 12.7** Aufbau einer interaktiven Fernsitzung



**HINWEIS:** Man kann sich in einer interaktiven Sitzung immer nur mit genau einem entfernten System verbinden. Man kann aber auf einem System mehrere PowerShell-Fenster öffnen und sich darin mit jeweils einem anderen entfernten System verbinden.

### Möglichkeiten einer Fernsitzung

In einer Fernsitzung kann jegliche Form von Änderungen durchgeführt werden, sowohl durch Ausführung von Commandlets, z. B.

```
(Get-service BITS) | start-service
```

als auch durch den Aufruf von Methoden

```
(Get-service BITS).Start()
```

Man kann auch Windows-Prozesse starten. Zu beachten ist jedoch, dass man auf dem entfernten System Benutzeroberflächen von diesen Prozessen nicht sehen kann, selbst wenn man dort lokal angemeldet ist.

Zum Verlassen der Fernsitzung gibt man ein:

```
Exit-PSSession
```

## ■ 12.7 Fernausführung von Befehlen

Um einen einzelnen Befehl auf einem entfernten System auszuführen, kann man auch das Commandlet `Invoke-Command` mit dem Parameter `-ComputerName` verwenden. Beim Parameter `-ScriptBlock` kann man einen oder mehrere (durch Zeilenumbruch oder Semikolon getrennte) Befehle angeben und auch Pipelines nutzen. Nicht nur Commandlets, auch klassische Kommandozeilenbefehle sind möglich.

### Listing 12.1 [2\_Aufbauwissen\Remoting\WPS2\_Remoting\_Script.ps1]

```
Invoke-Command -ComputerName F170 -scriptblock { Get-Service b* }
Invoke-Command -ComputerName F170 -scriptblock { Get-Service | sort status | ft name,
status }
Invoke-Command -computer F170 -Scriptblock { "Computername: " + [System.
Environment]::MachineName ; "Zeit: " + [DateTime]::Now ; "Sprache: " +
(Get-Culture) }
Invoke-Command -computer F170 -Scriptblock { ping www.it-visions.de }
```

Natürlich kann man den Skriptblock vorher auch in einer Variable speichern.

### Listing 12.2 [2\_Aufbauwissen\Remoting\WPS2\_Remoting\_Script.ps1]

```
$cmd = {
"Rechnername: $($[System.Environment]::MachineName)"
"Benutzername: $($[System.Environment]::UserDomainname + "\" + [System.
Environment]::Username)"
}
Invoke-Command PC199 -ScriptBlock $cmd
```

Ein Sonderfall ist gegeben, wenn der Skriptblock unterschiedlich für verschiedene Rechner sein soll. Dann muss der Skriptblock mit `[scriptblock]::Create()` erzeugt werden.

### Listing 12.3 [2\_Aufbauwissen\Remoting\Remoting\_DynamicScriptBlock.ps1]

```
$pcListe = "D140","D141"
foreach($pc in $pcListe)
{
"Abfrage bei $pc"
```

```

$scriptblock = [scriptblock]::Create( `
'"PC ' + $pc + ' nennt sich "' + `
'+ [System.Environment]::MachineName')

"Erzeugtes Skript..."
$scriptblock
"Skript wird gesendet..."
$ergebnis = Invoke-Command -ScriptBlock $scriptblock -ComputerName $pc
$ergebnis
}

```



**HINWEIS:** Alle Commandlets oder Anwendungen, die in dem Skriptblock gestartet werden, müssen auf dem Zielsystem verfügbar sein.

Auch hier wird das Ergebnis auf dem lokalen System angezeigt, oft mit der zusätzlichen Spalte „PSComputerName“, die den Namen des aufgerufenen Computers enthält.

```

Auswählen Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\hs> Get-Service b*

Status Name DisplayName
-----
Running BFE Basisfiltermodul
Running BITS Intelligenter Hintergrundübertragun...
Stopped Browser Computerbrowser

PS C:\Users\hs> Invoke-Command -ComputerName F170 -scriptblock { Get-Service b* }

Status Name DisplayName PSComputerName
-----
Stopped BDESUC BitLocker Drive Encryption Service f170
Running BFE Base Filtering Engine f170
Stopped BITS Background Intelligent Transfer Ser... f170
Running Browser Computer Browser f170
Stopped bthserv Bluetooth Support Service f170

PS C:\Users\hs>

```

**Bild 12.8** Lokaler Aufruf versus entfernter Aufruf

Allerdings muss man beachten, dass die Ergebnismenge keineswegs die gleiche Struktur wie bei einem lokalen Aufruf hat. Die Objekte in der Pipeline sind nicht vom Typ `System.ServiceProcess.ServiceController`, sondern `Deserialized.System.ServiceProcess.ServiceController`. Zwischen den Rechnergrenzen hat, für den Transport über das Netzwerk, eine Serialisierung/Deserialisierung der Objekte stattgefunden. Dabei sind die Methoden der Objekte „verloren“ gegangen. Methodenaufrufe wie

```
(Invoke-Command -ComputerName F170 -scriptblock { Get-Service bits }).Start()
```

sind also nicht möglich!

```

Administrator: Windows PowerShell

PS C:\Users\hs> Get-Service b* | gm

    TypeName: System.ServiceProcess.ServiceController

Name           MemberType      Definition
-----
Name           AliasProperty   Name = ServiceName
RequiredServices AliasProperty   RequiredServices = ServicesDependedOn
Disposed       Event            System.EventHandler Disposed(System.Object, System.EventArgs)
Close          Method          System.Void Close()
Continue       Method          System.Void Continue()
CreateObjRef   Method          System.Runtime.Remoting.ObjRef CreateObjRef(type requestedType)
Dispose        Method          System.Void Dispose()
Equals         Method          bool Equals(System.Object obj)
ExecuteCommand Method          System.Void ExecuteCommand(int command)
GetHashCode    Method          int GetHashCode()
GetLifetimeService Method        System.Object GetLifetimeService()
GetType        Method          Type GetType()
InitializeLifetimeService Method        System.Object InitializeLifetimeService()
Pause          Method          System.Void Pause()
Refresh        Method          System.Void Refresh()
Start          Method          System.Void Start(), System.Void Start(string[] args)
Stop           Method          System.Void Stop()
ToString       Method          string ToString()
WaitForStatus Method          System.Void WaitForStatus(System.ServiceProcess.ServiceControllerStatus des...
CanPauseAndContinue Property        System.Boolean CanPauseAndContinue {get;}
CanShutdown   Property        System.Boolean CanShutdown {get;}
CanStop       Property        System.Boolean CanStop {get;}
Container      Property        System.ComponentModel.IContainer Container {get;}
DependentServices Property        System.ServiceProcess.ServiceController[] DependentServices {get;}
DisplayName    Property        System.String DisplayName {get;set;}
MachineName    Property        System.String MachineName {get;set;}
ServiceHandle  Property        System.Runtime.InteropServices.SafeHandle ServiceHandle {get;}
ServiceName    Property        System.String ServiceName {get;set;}
ServicesDependedOn Property        System.ServiceProcess.ServiceController[] ServicesDependedOn {get;}
ServiceType    Property        System.ServiceProcess.ServiceType ServiceType {get;}
Site           Property        System.ComponentModel.Site Site {get;set;}
Status         Property        System.ServiceProcess.ServiceControllerStatus Status {get;}

PS C:\Users\hs> Invoke-Command -ComputerName F170 -scriptblock { Get-Service b* } | gm

    TypeName: Deserialized.System.ServiceProcess.ServiceController

Name           MemberType      Definition
-----
ToString       Method          string ToString(), string ToString(string format, System.IFormatProvider formatProv...
Name           NoteProperty    System.String Name=BDESUC
PSComputerName NoteProperty    System.String PSComputerName=f170
PSShowComputerName NoteProperty    System.Boolean PSShowComputerName=True
RequiredServices NoteProperty    Deserialized.System.ServiceProcess.ServiceController[] RequiredServices=
RunspaceId     NoteProperty    System.Guid RunspaceId=c44b56db-fd2f-4769-add-2cd985a932f2
CanPauseAndContinue Property        System.Boolean {get;set;}
CanShutdown    Property        System.Boolean {get;set;}
CanStop        Property        {get;set;}
Container       Property        Deserialized.System.ServiceProcess.ServiceController[] {get;set;}
DependentServices Property        System.ServiceProcess.ServiceController[] {get;set;}
DisplayName     Property        System.String {get;set;}
MachineName     Property        System.String {get;set;}
ServiceName     Property        System.String {get;set;}
ServicesDependedOn Property        Deserialized.System.ServiceProcess.ServiceController[] {get;set;}
ServiceType     Property        System.String {get;set;}
Site            Property        {get;set;}
Status          Property        System.String {get;set;}

PS C:\Users\hs> ^W_

```

Bild 12.9 Get-Member nach einem lokalen und einem entfernten Aufruf von Get-Service



**ACHTUNG:** Beim Pipelining kann man in eine sehr tiefe Falle tappen. Ein Benutzer, der merkt, dass

```
(Invoke-Command -ComputerName F170 -scriptblock { Get-Service Bits
}).Start()
```

nicht funktioniert, würde wohl auf

```
(Invoke-Command -ComputerName F170 -scriptblock { Get-Service Bits }) |
Start-Service
```

ausweichen wollen.

Dieser Befehl würde ohne Fehlermeldung abgeschlossen – er hätte aber nicht getan, was gewünscht war. In diesem Fall würde der „BITS“-Dienst auf dem lokalen System, nicht auf dem entfernten System gestartet. Der Grund liegt darin, dass Start-Service das PSComputerName-Attribut ignoriert und nur den Namen des Dienstes berücksichtigt. Die folgende Bildschirmabbildung liefert den Beweis.



```

PS C:\Users\hs> Invoke-Command -Session $s -ScriptBlock { Get-Service BITS }
Status Name DisplayName PSComputerName
-----
Stopped BITS Background Intelligent Transfer Ser... f170

PS C:\Users\hs> Get-Service BITS
Status Name DisplayName
-----
Stopped BITS Intelligenter Hintergrundübertragun...

PS C:\Users\hs> Invoke-Command -Session $s -ScriptBlock { Get-Service BITS } | Start-Service
PS C:\Users\hs> Invoke-Command -Session $s -ScriptBlock { Get-Service BITS }
Status Name DisplayName PSComputerName
-----
Stopped BITS Background Intelligent Transfer Ser... f170

PS C:\Users\hs> Get-Service BITS
Status Name DisplayName
-----
Running BITS Intelligenter Hintergrundübertragun...

```

**Bild 12.10** Unerwartetes Verhalten beim Fernaufruf

Viele Commandlets funktionieren so, auch weitreichende wie `Remove-Item`. Der folgende Befehl löscht also nicht Textdateien auf dem entfernten, sondern gleichnamige Dateien auf dem lokalen System!

```
Invoke-Command -ComputerName F170 -scriptblock { Get-Item d:\Daten\*.txt } | remove-Item
```

Richtig ist hier, den Befehl `Remove-Item` mit in den Skriptblock zu nehmen:

```
Invoke-Command -ComputerName F170 -scriptblock { Get-Item d:\Daten\*.txt | remove-Item }
```

Grund für diese Falle ist, dass die von `Get-Item` gelieferten Dateiobjekte zwar von der PowerShell um den Parameter `PSComputerName` angereichert wurden, aber das Commandlet `Remove-Item` diese Zusatzinformation leider ignoriert.



**HINWEIS:** Einige Commandlets in der PowerShell, darunter `Get-Process` und `Get-Service`, bieten auch noch einen kürzeren Weg für die Fernabfrage an. Bei diesen Commandlets kann man ein einzelnes entferntes System über den Parameter `-Computer` angeben, z. B. `Get-Process -Computer F111` (Details siehe Kapitel „Fernausführung/Fernabfrage ohne WS-Management“).

Vorteil dieser Methode ist, dass man dafür nicht WS-Management braucht, also auch ältere Betriebssysteme abfragen kann, für die es kein WS-Management gibt. Nachteil ist, dass sich die Fernabfrage immer nur auf den einzelnen Befehl bezieht. Man kann weder Befehlsfolgen noch Skripte angeben. Außerdem kann man immer nur ein einzelnes entferntes System ansprechen.

Noch ein Tipp: Sie können auch einen Skriptblock mit Zeilenumbrüchen an der Konsole erfassen. Dazu müssen Sie aber innerhalb des Skriptblocks `SHIFT + EINGABE` drücken. Erst wenn Sie den Skriptblock mit der schließenden geschweiften Klammer beendet haben, drücken Sie nur noch `EINGABE`. Wenn Sie vorzeitig nur auf `EINGABE` drücken, wird der Befehl sofort ausgeführt.

```
PS T:\> invoke-command -computer D142 -ScriptBlock {
>>> if (-not (test-path c:\temp)) { md c:\temp; "Temp angelegt!" }
>>> dir c:\
>>> }

Verzeichnis: C:\

Mode                LastWriteTime         Length Name                                           PSComputerName
-----
d-----            19.01.2016         17:16         temp                                           D142
Temp angelegt!
d-----            22.08.2013         17:52         PerfLogs                                       D142
d-r-----           22.12.2015         22:00         Program Files                                 D142
d-----            19.01.2016         16:45         Program Files (x86)                          D142
d-----            19.01.2016         17:16         temp                                           D142
d-r-----            06.12.2015         19:53         Users                                         D142
d-----            22.12.2015         21:20         Windows                                       D142

PS T:\> _
```

**Bild 12.11** Eingabe eines Skriptblocks mit Umbrüchen an der Konsole

## ■ 12.8 Parameterübergabe an die Fernausführung

Oft braucht der auf dem entfernten System auszuführende Skriptblock Werte, die der aufrufende Computer besitzt. Das erste Beispiel zeigt, wie es falsch ist. `$VM` und `$Path` sind leer bei der Ausführung auf Server79.

```
$VM = "D140"
$Path = "\\D123\backup"

Invoke-Command -ComputerName Server79 -ScriptBlock {
# falsch. $VM ist leer !!!
"Sichern der VM $vm nach $Path..."
}
```

Richtig ist, den Inhalt von `$VM` und `$Path` per Parameter `-ArgumentList` an `Invoke-Command` zu übergeben und dann in dem fernauszuführenden Skriptblock darauf mit `$args[0]` und `$args[1]` zuzugreifen.

```
$VM = "D140"
$Path = "\\D123\backup"
#richtig!
Invoke-Command -ComputerName Server79 -ArgumentList $vm,$Path -ScriptBlock {
$VM = $args[0]
$Path = $args[1]
"Sichern der VM $vm nach $Path..."
}
```

## ■ 12.9 Fernausführung von Skripten

Mit Invoke-Command kann man natürlich ein auf dem entfernten System vorhandenes Skript starten, z.B.:

```
Invoke-Command -computer F170 -scriptblock { d:\Skripte\WPS2_Computername.ps1 }
```

Voraussetzung ist natürlich, dass auf dem entfernten System die Skriptausführung erlaubt ist und alle für das Skript benötigten Dateien dort sind.

```
PS T:\> invoke-command -computer d140 -ScriptBlock { c:\skripte\Benutzeranlegen.ps1 }
File C:\skripte\Benutzeranlegen.ps1 cannot be loaded because running scripts is disabled on this system. For more
information, see about_Execution_Policies at http://go.microsoft.com/fwlink/?linkID=135170.
+ CategoryInfo          : Sicherheitsfehler: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
+ PSComputerName        : d140
PS T:\>
```

**Bild 12.12** Fehlermeldung, wenn das Starten des Skripts auf einem entfernten System nicht erlaubt ist

Mit folgendem Befehl kann man die Skriptausführung auf einem entfernten System aktivieren:

```
Invoke-Command -computer F170 -scriptblock { Set-executionpolicy unrestricted }
```

Das folgende Listing zeigt ein Beispiel, bei dem man die Skriptausführung aktiviert, ein Skript kopiert, dann ausführt und anschließend das Skript löscht und die Skriptausführung wieder deaktiviert.

**Listing 12.4** Entfernte Skriptausführung durch Skriptkopieren

[2\_Aufbauwissen/Remoting/WPS2\_Remoting\_Script.ps1]

```
$computer = "PC170"
$lokalesSkript = "H:\TFS\Demos\PowerShell\2_Aufbauwissen\Remoting\WPS2_Computername.
ps1"
"Start Session..."
$s = New-PSSession -ComputerName $computer

"Enable Script Execution on remote System..."
Invoke-Command -Session $s -scriptblock { Set-executionpolicy unrestricted }

"Copy Script..."
Copy-Item $lokalesSkript "\\$computer\c$\temp\wps2_Computername.ps1"

"Start Script..."
Invoke-Command -Session $s -scriptblock { c:\temp\WPS2_Computername.ps1 }

"Delete Script..."
Remove-Item "\\$computer\c$\temp\wps2_Computername.ps1"

"Disable Script Execution on remote System..."
Invoke-Command -Session $s -scriptblock { Set-executionpolicy default }

"End Session!"
Remove-PSSession $s
```

Das gleiche Ergebnis kann man aber auch viel einfacher haben, denn das Commandlet `Invoke-Command` bietet auch die Möglichkeit, ein lokales Skript auf den entfernten Computer zu übertragen und dort zu starten:

```
Invoke-Command -computer PC170 -FilePath H:\WPS2_Computername.ps1
```

## ■ 12.10 Ausführung auf mehreren Computern

Das Commandlet `Invoke-Command` bietet auch die Möglichkeit, mehrere Computer in Form eines Arrays (eine durch Komma getrennte Liste) anzugeben.

**Beispiel 1:** Setzen von Datum und Uhrzeit auf mehreren Computern:

```
Invoke-Command -computer F170, F171, F172, F173 -Script { Set-date -date }
```

```
PS C:\> Invoke-Command -computer F170, F171, F172, F173 -Script { set-date -date "24.6.2009 15:10:00"
-cred $cred
Mittwoch, 24. Juni 2009 15:10:00
Mittwoch, 24. Juni 2009 15:10:00
Mittwoch, 24. Juni 2009 15:10:00
Mittwoch, 24. Juni 2009 15:10:00
```

**Bild 12.13** Ausführen von Set-Date auf mehreren Computern

**Beispiel 2:** Auslesen von `c:\Temp` auf mehreren Computern

```
$computer = F170, F171, F172, F173, F174
Invoke-Command -computer $computer { Get-childitem c:\temp }
```



**TIPP:** Für den lokalen Computer kann man „localhost“ oder „.“ verwenden.

Das Ergebnis ist eine Gesamtliste der Ergebnisse von allen genannten Computern. Die Objekte in der Menge besitzen ein zusätzliches Attribut (NoteProperty) `PSComputerName`, das den Namen des Computers zeigt, der das Objekt geliefert hat. Dadurch ist ein Filtern/Sortieren/Gruppieren auf dem aufrufenden Computer möglich.



**HINWEIS:** Die PowerShell fragt bei den einzelnen Computern in der Reihenfolge an, wie sie im Array angegeben sind. Die Reihenfolge der Ergebnisse ist jedoch abhängig davon, wann die Ergebnisse eintreffen.

```
PS C:\Users\HS> invoke-command -computer F171,F172,F173 -scriptblock { get-service i* } | ft name, status, PSComputerName
```

Name	Status	PSComputerName
idsvc	Stopped	f173
IREST	Running	f173
IPBusEnum	Stopped	f173
iphlpvc	Running	f173
idsvc	Stopped	f171
IREST	Stopped	f171
IPBusEnum	Stopped	f171
iphlpvc	Running	f171
idsvc	Stopped	f172
IREST	Running	f172
IPBusEnum	Stopped	f172
iphlpvc	Running	f172

```
PS C:\Users\HS>
```

**Bild 12.14** Ergebnismenge eines Abrufs von Dienst-Objekten von drei Computern



**TIPP:** Bei der Ausgabe von einigen Klassen (z. B. auch bei ServiceController) wird PSComputerName automatisch ausgegeben. Dies kann man durch `-Hide-Computername` unterdrücken.

```
Invoke-Command -Computer F170, F171 -ScriptBlock { Get-Service I* }
-HideComputername
```

## Abbrechen eines Fernbefehls

Zum Abbrechen eines entfernt ausgeführten Befehls kann man wie bei lokalen Befehlen die Tastenkombination **STRG + C** verwenden.

## 12.11 Sitzungen

Invoke-Command erzeugt im Standard eine temporäre Verbindung. Alle Definitionen (Variablen und Funktionen), die im Rahmen der Ausführung von Invoke-Command auf einem entfernten System erzeugt wurden, sind nach Ende des Befehls wieder ungültig.

Die Alternative ist eine permanente Verbindung (Sitzung). Eine Sitzung (engl. Session, alias „PSSession“) ist Host für die PowerShell, in der die PowerShell Befehle ausführt. Aus der Sicht von Windows ist eine Session ein Prozess.

Eine Sitzung gestaltet man durch das Laden von Snap-Ins und Modulen sowie durch die Definition von Variablen, Funktionen und Aliasen. Alle diese Einstellungen leben so lange, wie die Sitzung dauert.

Beim Start der PowerShell durch PowerShell.exe wird automatisch eine Sitzung erzeugt (Default Session). Durch Commandlets kann man weitere Sitzungen auf dem lokalen Computer oder entfernten Computern erzeugen.



**HINWEIS:** Alle Fernaufrufe der PowerShell erfolgen im Rahmen einer Sitzung. Die PowerShell unterscheidet temporäre Sitzungen (mit Invoke-Command unter Angabe eines Computernamens) und permanente Sitzungen (mit Invoke-Command unter Angabe eines Sitzungsobjekts, das vorher mit `New-PSSession` erzeugt wurde).

## Commandlets zur Sitzungsverwaltung

Es folgt ein Überblick über die Commandlets zur Sitzungsverwaltung:

- `New-PSSession`: Erzeugen einer neuen Sitzung auf dem lokalen oder einem entfernten Computer
- `Get-PSSession`: Liste aller Sitzungen, die aus der aktuellen Sitzung heraus gestartet wurden (zeigt aber nicht Sitzungen, die andere Computer auf dem lokalen Computer geöffnet haben)
- `Remove-PSSession`: entfernt eine Session oder alle Sessions (`Remove-PSSession *`)
- `Enter-PSSession`: Start einer interaktiven Sitzung auf dem lokalen oder einem entfernten Computer
- `Exit-PSSession`: Ende einer interaktiven Sitzung
- `Disable-PSSessionConfiguration`: Sperren einer/aller Sitzungskonfigurationen
- `Enable-PSSessionConfiguration`: Entsperren einer/aller Sitzungskonfigurationen
- `Get-PSSessionConfiguration`: Auflisten der Sitzungskonfigurationen
- `Register-PSSessionConfiguration`: permanente Registrierung einer Sitzungskonfiguration
- `Set-PSSessionConfiguration`: Setzen von Eigenschaften einer Sitzungskonfiguration
- `Unregister-PSSessionConfiguration`: Löschen einer Sitzungskonfiguration

## Sitzungen erstellen

Eine Sitzung erzeugt man über `New-PSSession`:

```
$s = New-PSSession -computername F171
```

Auf diese offene Sitzung muss man im Commandlet `Invoke-Command` Bezug nehmen. Ein Computernamen ist dann nicht mehr erforderlich.

```
Invoke-Command -session $s -scriptblock {$p = Get-Process }
```

Man kann auch permanente Sitzungen zu mehreren Computern aufbauen, indem man bei `New-PSSession` mehrere Computer angibt:

```
$s = New-PSSession -computername F173, D144, D145
Invoke-Command -session $s -scriptblock {Get-culture}
```

Alternativ kann man auch mehrere einzelne Sitzungen erstellen und diese bei `Invoke-Command` angeben.

```
"Sitzungen erstellen..."
$s1 = New-PSSession -ComputerName F173
$s2 = New-PSSession -ComputerName D144
$s3 = New-PSSession -ComputerName D145

"Fernzugriff auf alle drei Rechner..."
Invoke-Command -Session $s1, $s2, $s3 -ScriptBlock { Get-Service spooler }
```

Die PowerShell-Profilskripte werden weder in temporären noch in permanenten Verbindungen automatisch auf dem entfernten System geladen. Bei Bedarf müssen sie explizit gestartet werden, z. B.:

```
Invoke-Command -session $s { . "$home\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1" }
```

## Schließen von Sitzungen

Sitzungen werden automatisch geschlossen, wenn die Sitzung (Elternsitzung) beendet wird, aus der heraus die Sitzung (Kindsitzung) gestartet wurde. Entfernte Sitzungen enden zudem automatisch, wenn entfernte Computer für vier Minuten lang nicht mehr erreichbar sind. Manuell kann man eine Sitzung mit `Remove-PSSession` schließen.

## Sitzungskonfigurationen

Eine Sitzungskonfiguration legt durch zahlreiche Einstellungen fest, wer eine Sitzung aufbauen darf und welche Befehle in der Sitzung zur Verfügung stehen. Typische Einstellungen sind:

- Benutzer, die sich mit dem Computer entfernt verbinden dürfen
- Größe der Objekte, die die entfernten Benutzer übertragen dürfen
- Festlegung der verfügbaren Commandlets und Funktionen



**HINWEIS:** Die Konfiguration einer Sitzung ist nur möglich, wenn die PowerShell als Administrator gestartet wurde.

Die Standardkonfiguration trägt den Namen „Microsoft.PowerShell“. Auf 64-Bit-Computern gibt es zusätzlich „Microsoft.PowerShell32“. Seit Windows Server 2008 R2 gibt es außerdem „Microsoft.ServerManager“.

Die verfügbaren Konfigurationen mit zahlreichen Details zeigt:

```
Get-PSSessionconfiguration | fl
```

oder alternativ

```
dir wsman:localhost/plugin
```

## Zugriffsrechte für Fernaufrufe

Im Standard können nur Administratoren Fernaufrufe ausführen. Man kann aber die Zugriffsrechteliste (Access Control List – ACL) der Sitzungskonfiguration ändern.

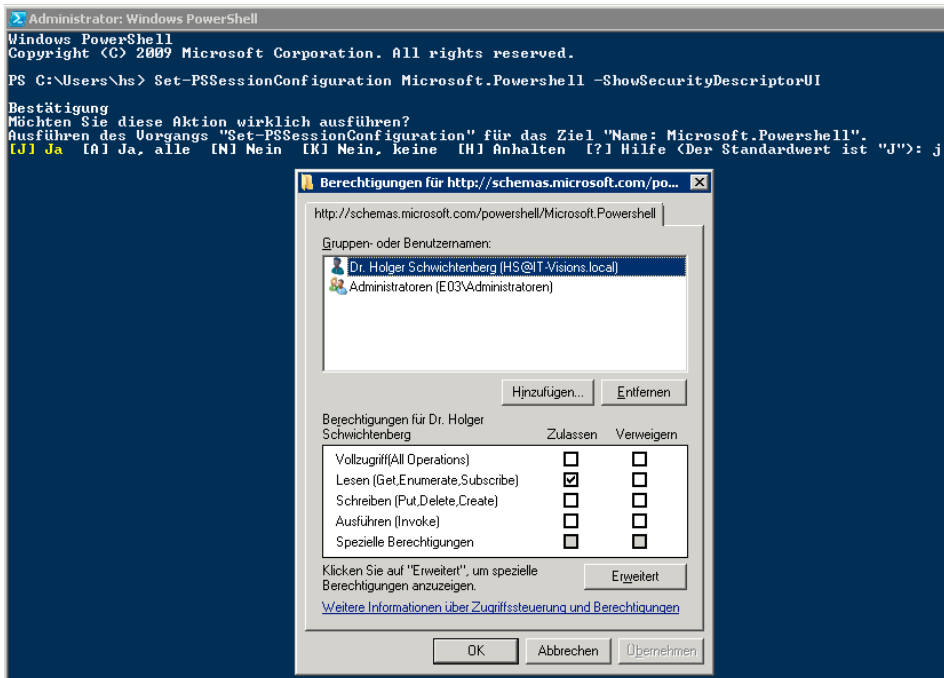
```
Set-PSSessionConfiguration Microsoft.PowerShell -ShowSecurityDescriptorUI
```

Etwas kurios für eine kommandozeilenbasierte Shell ist, dass sich dadurch ein Windows-Fenster öffnet, wie man es von den Zugriffsrechtelisten von Windows Explorer und der Windows-Registrierungsdatenbank kennt. Rein kommandozeilenbasiert kann man arbei-

ten, indem man im Parameter `SecurityDescriptorSDDL` eine SDDL-Zeichenkette (SDDL = Security Descriptor Definition Language) übergibt.



**HINWEIS:** Das Commandlet `Disable-PSSessionConfiguration` verändert die Zugriffsrechtelisten für eine oder alle vorhandenen Sitzungskonfigurationen so, dass kein Benutzer mehr Rechte für den Fernaufruf hat.



**Bild 12.15** Ändern der ACL für die PowerShell-Sitzungskonfiguration

### Beispiel:

```
disable-PSSessionConfiguration -name Microsoft.PowerShell
```

Die Blockade kann man rückgängig machen mit `Enable-PSSessionConfiguration`.

Mit `Register-PSSessionConfiguration` kann man eine neue Konfiguration erzeugen. Diese wird permanent auf dem Computer gespeichert. Der erforderliche Neustart des WS-Management-Dienstes wird auf Nachfrage ausgeführt.

Der folgende Befehl legt eine neue Konfiguration unter dem Namen „FBIconfig“ an mit einer Erhöhung der Datenmenge auf 200 MB (Standard sind 50 MB) und einem Skript, das beim Starten der Sitzung ausgeführt werden soll. Das Skript muss lokal auf dem System vorhanden sein.

```
register-PSSessionConfiguration -name FBIconfig  

-MaximumReceivedDataSizePerCommandMB 200  

-StartupScript h:\Skripte\WPS2_Remoting_SessionStartSkript.ps1
```

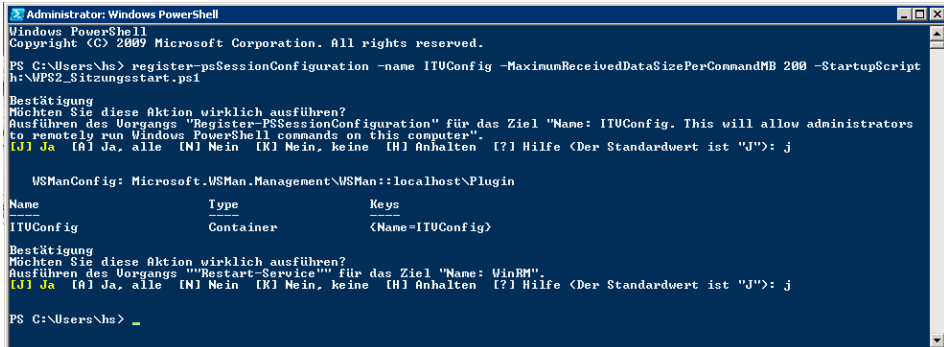


Anschließend kann man eine Sitzung mit Bezug auf diese Konfiguration starten:

```
$s = New-PSSession -ConfigurationName FBIConfig
```

Auch Invoke-Command hat den Parameter -ConfigurationName:

```
Invoke-Command -ConfigurationName "FBIConfig" -scriptblock { Start-Service BITS }  
-computer F173
```



**Bild 12.16** Ablauf der Registrierung einer Sitzungskonfiguration

In `$PSSessionConfigurationName` ist der Name der Konfiguration abgelegt, die verwendet wird, wenn man den Parameter `-ConfigurationName` nicht angibt. Den Inhalt dieser Variablen kann man ändern.

Zum Löschen einer Konfiguration verwendet man:

```
Unregister-PSSessionConfiguration -name FBIConfig
```

## ■ 12.12 Implizites Remoting

Implizites Remoting ist eine weitere Form des PowerShell Remoting, bei der für eine entfernt auszuführende PowerShell-Funktion ein Commandlet auf dem Client als Alias verwendet wird.

Man erstellt eine PowerShell-Remoting-Session:

```
$session = New-PSSession D140
```

Man sendet dann in der Sitzung eine Funktion zu dem entfernten System. Ausgeführt wird dadurch dort nichts.

```
Invoke-Command $session -scriptblock { function Get-ComputerName { [System.  
Environment]::MachineName }
```

Nun kann man diese Funktion auf dem entfernten System im Rahmen der Sitzung aufrufen:

```
Invoke-Command $session {Get-ComputerName}
```

Alternativ kann man – via `Import-PSSession` – das implizite Remoting einrichten. Dadurch entsteht ein lokales Alias-Commandlet, das aber nicht lokal ausgeführt wird, sondern im Rahmen der Sitzung zum Fernsystem gesendet wird:

```
Import-PSSession -Session $session -CommandName Get-ComputerName
```

Die oben gesendete Funktion auf dem fernen System kann nun so gestartet werden, ohne Angabe der Sitzung oder des Namens des Fernsystems:

```
Get-ComputerName
```



**ACHTUNG:** Der Einsatz von implizitem PowerShell Remoting kann nicht zu den empfehlenswerten Vorgehensweisen („Best Practices“) zählen, weil man dem Commandlet nicht ansehen kann, wo es ausgeführt wird. Das kann zu Unklarheiten führen und zu langsamen Ausführungszeiten, die nicht direkt erkennbar sind.

## ■ 12.13 Zugriff auf entfernte Computer außerhalb der eigenen Domäne

Der Zugriff auf entfernte Computer außerhalb der eigenen bzw. einer vertrauenden Domäne ist möglich.

Grundsätzlich gibt es zwei Möglichkeiten für den domänenübergreifenden Zugriff:

- Expliziter Eintrag des Zielsystems in die Liste vertrauter Systeme
- Einrichten von Secure Socket Layer (SSL) für die HTTP-Kommunikation alias HTTPS

### Herleitung des Problems

Wenn man einen domänenübergreifenden Aufruf versucht (z. B. `Enter-PSSession F171`), wird man auf folgende Fehler stoßen:

*„Die Anforderung kann von WinRM nicht verarbeitet werden. Bei Verwendung der Kerberos-Authentifizierung ist der folgende Fehler aufgetreten: Der Netzwerkpfad wurde nicht gefunden. Mögliche Ursachen:*

*Der angegebene Benutzername oder das angegebene Kennwort ist ungültig. – Kerberos wird verwendet, wenn keine Authentifizierungsmethode und kein Benutzername angegeben werden.*

*Kerberos akzeptiert Domänenbenutzernamen, aber keine lokalen Benutzernamen.*

*Der Dienstprinzipalname (Service Principal Name, SPN) für den Remotecomputernamen und -port ist nicht vorhanden.*

*Der Clientcomputer und der Remotecomputer befinden sich in unterschiedlichen Domänen, zwischen denen keine Vertrauensbeziehung besteht.*

*Wenn Sie die oben genannten Ursachen überprüft haben, probieren Sie folgende Aktionen aus: - Suchen Sie in der Ereignisanzeige nach Ereignissen im Zusammenhang mit der Authentifizierung.*

*Ändern Sie die Authentifizierungsmethode; fügen Sie den Zielcomputer der Konfigurationseinstellung ‚TrustedHosts‘ für WinRM hinzu, oder verwenden Sie den HTTPS-Transport. Beachten Sie, dass Computer in der TrustedHosts-Liste möglicherweise nicht authentifiziert sind.“*

Dieser ausführliche Fehlertext liefert schon recht genaue Hinweise auf das Problem: Die Authentifizierungsmethode Kerberos funktioniert nur in Domänen. Es ist also eine andere Authentifizierungsmethode zu wählen.

Verfügbare Standardauthentifizierungsmethoden sind Default (= Kerberos), Basic, Negotiate (= Aushandlung zwischen Client und Server mit dem Simple and Protected GSSAPI Negotiation Mechanism – SPNEGA), NegotiateWithImplicitCredential, Credssp, Digest, Kerberos.

Ein neuer Versuch könnte dann also Digest oder Basic sein, also:

```
Enter-PSSession F171 -Authentication Digest
```

oder

```
Enter-PSSession F171 -Authentication Basic
```

Nun lautet der Fehlertext in beiden Fällen:

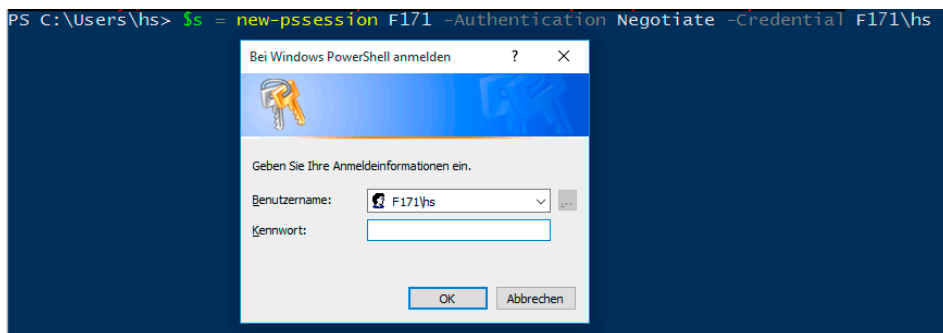
*„Der WinRM-Client kann die Anforderung nicht verarbeiten. Wenn der Basic- oder Digest-Authentifizierungsmechanismus verwendet wird, müssen Anforderungen den Benutzernamen und das Kennwort enthalten. Fügen Sie den Benutzernamen oder das Kennwort hinzu, oder ändern Sie den Authentifizierungsmechanismus, und wiederholen Sie die Anforderung.“*

Dies bedeutet also, dass die Daten des am lokalen System angemeldeten Benutzers nicht automatisch übermittelt werden (selbst wenn es auf dem Zielsystem eine gleiche Kombination aus Benutzername und Kennwort gibt).

Nun ein dritter Versuch, wobei man Client und Server die Authentifizierung aushandeln lässt:

```
Enter-PSSession F171 -Authentication Negotiate -credential F171\hs
```

Die PowerShell zeigt den Authentifizierungsdialog ...



**Bild 12.17** Authentifizierungsdialog

und dann wieder einen Fehler:

*Der WinRM-Client kann die Anforderung nicht verarbeiten. Wenn das Authentifizierungsschema nicht Kerberos ist oder der Clientcomputer nicht Mitglied einer Domäne ist, muss der HTTPS-Datentransport verwendet werden, oder der Zielcomputer muss der TrustedHosts-Konfigurationseinstellung hinzugefügt werden. Verwenden Sie winrm.cmd, um TrustedHosts zu konfigurieren. Beachten Sie, dass Computer in der TrustedHosts-Liste möglicherweise nicht authentifiziert sind.*

Auch hier ist die Anweisung klar: Entweder ist HTTPS zu verwenden oder aber ein Eintrag in TrustedHosts vorzunehmen. Letzteres ist einfacher.

### Eintrag in die Liste vertrauter Systeme

Diesen Eintrag kann man über den WSMAN-Navigationsprovider der PowerShell recht elegant auf dem Client vornehmen:

```
cd WSMAN:\localhost\Client
Set-Item trustedhosts "F170, F171, F172, F173, F174, F175" -force
Restart-Service winrm
```

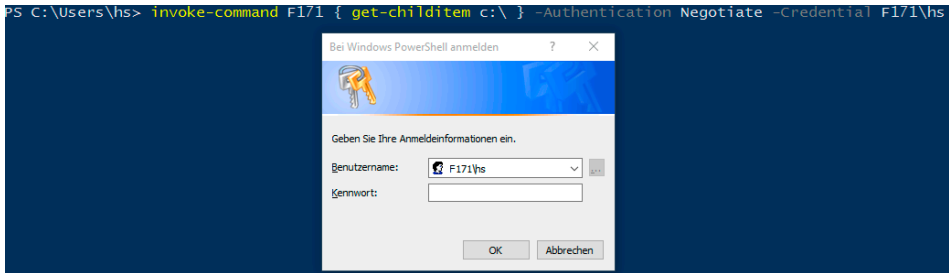
Alternativ können auch IP-Adressen statt Rechnernamen angegeben werden.

Die Veränderungen der Eigenschaft „TrustedHost“ erfordern eigentlich eine Rückbestätigung, daher das `-force`. TrustedHost ist eine Liste der zu vertrauenden Rechner (Rechnername oder IP-Adresse). Das Skript muss mit Administratorrechten gestartet werden.

### Ausführen von Befehlen auf entfernten Computern außerhalb der eigenen Domäne

Danach kann man dann mit einem Befehl wie folgt einen Fernzugriff ausführen, wobei die PowerShell explizit nach Benutzername und Kennwort fragen wird, selbst wenn auf dem Zielsystem die gleiche Benutzername-Kennwort-Kombination existiert.

```
Invoke-Command F171 { Get-ChildItem c:\ } -authentication negotiate -credential F171\hs
```



**Bild 12.18** Nachfrage der PowerShell beim Zugriff auf Computer, die nicht zur (vertrauten) Domäne gehören

Man kann auch eine dauerhafte Sitzung zu einem entfernten Computer, der nicht zur Domäne gehört, erstellen:

```
$s = New-PSSession F171 -authentication negotiate -credential F171\hs
```



**TIPP:** Die Anmeldedaten immer wieder eingeben zu müssen, kann man vermeiden, indem man sich die Anmeldedaten in einer Variablen merkt:

```
$cred = Get-Credential
```

Hinweis: Man muss die Anmeldedaten nicht pro Computer eingeben. Sofern Benutzername und Kennwort auf allen Systemen gleich sind, kann man auch Folgendes schreiben:

**Listing 12.5** Setzen des Datums mit Set-Date auf mehreren Nicht-Domänencomputern, die die gleiche Benutzername-Kennwort-Kombination haben [WPS2\_Remoting\_MultiComputer\_SetDate.ps1]

```
$cred = Get-Credential
$s = New-PSSession -auth negotiate -cred $cred -computer F170, F171, F171,
F173, F174
Invoke-Command -Script { Set-date -date "24.6.2009 15:20:00" } -session $s
```

## ■ 12.14 Verwaltung des WS-Management-Dienstes

In PowerShell ist der WS-Management-Dienst (WS-Man) über einen PowerShell-Provider (mit Namen „WSMan“) administrierbar. Im Standard erscheint in der Liste das Laufwerk „WSMan“.

```
PS C:\Users\hs> get-psdrive
```

Name	Used (GB)	Free (GB)	Provider	Root	CurrentLocation
A			FileSystem	A:\	
Alias			Alias		
C	10.90	86.65	FileSystem	C:\	Users\hs
cert			Certificate	\	
Env			Environment		
Function			Function		
G			FileSystem	G:\	
H	.09	97.57	FileSystem	H:\	
HKCU			Registry	HKCU_CURRENT_USER	
HKLM			Registry	HKLM_LOCAL_MACHINE	
S	5.24	227.65	FileSystem	S:\	
T	.09	37.48	FileSystem	T:\	
Variable			Variable		
WSMan			WSMan		

**Bild 12.19** Liste der PowerShell-Provider in PowerShell

In diesem Laufwerk kann man wie bei anderen Providern mit den Standard-Navigations-Commandlets wie `Get-ChildItem (dir)` und `Get-Item/Set-Item` arbeiten.

```
PS C:\Users\hs>
PS C:\Users\hs> cd wsman:\localhost
PS WSMan:\localhost> dir
```

WSManConfig: Microsoft.WSMan.Management\WSMan::localhost		
Name	Value	Type
MaxEnvelopeSizekb	150	System.String
MaxTimeouts	60000	System.String
MaxBatchItems	32000	System.String
MaxProviderRequests	4294967295	System.String
Client		Container
Service		Container
Shell		Container
Listener		Container
Plugin		Container
ClientCertificate		Container

**Bild 12.20** Auflisten von WSMan:/localhost

Im Standard erscheint unterhalb der Laufwerkswurzel WSMan: nur „localhost“. Durch das Commandlet `Connect-WSMan` unter Angabe eines Computernamens kann man hier aber weitere Computer integrieren und im Folgenden ansteuern.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\hs> dir wsman:

WSManConfig:
-----
ComputerName      Type
-----
localhost         Container

PS C:\Users\hs> connect-wsman -ComputerName F170
PS C:\Users\hs> dir wsman:

WSManConfig:
-----
ComputerName      Type
-----
localhost         Container
F170              Container

PS C:\Users\hs> ^R_
```

**Bild 12.21**  
Einsatz von `Connect-WSMan`

Die folgende Befehlsfolge zeigt das Setzen der vertrauenswürdigen Computer, zu denen eine Authentifizierung mit Basisauthentifizierung oder Digest toleriert werden soll.

```
cd WSMan:\localhost\Client
Set-Item trustedhosts "F170, F171, F172, F173, F174, F175" -force
Restart-Service winrm
```

Über das WSMAN-Laufwerk kann man auch den verwendeten Port ändern. Den aktuell verwendeten Port listet man auf mit:

```
Dir wsman:\localhost\listener\*\Port
```

Auf dem Zielsystem ändert man den Port mit (hier wird geändert auf Port 80):

```
set-item wsman:\localhost\listener\*\Port -value 80 -Force
```

Auf dem Client kann man dann den Port auf Aufruf angeben:

```
Enter-PSSession PC199 -Port 80
```

## ■ 12.15 PowerShell Direct für Hyper-V

Ein virtuelles System können Sie grundsätzlich genauso wie ein physikalisches System fernverwalten über RPC oder PowerShell-Remoting. Für PowerShell-Remoting muss dies auf dem Zielsystem aktiviert sein.

Ab Windows 10 und Windows Server 2016 bietet Microsoft für in Hyper-V gehostete virtuelle Maschinen eine Vereinfachung unter dem Namen „PowerShell Direct“ an. Hier ist eine PowerShell-Remoting-Verbindung nicht mehr notwendig. Auch die Firewall muss nicht geöffnet sein. Die PowerShell redet direkt über den VMBus von Hyper-V.

Voraussetzungen sind:

- Der Hyper-V-Host ist Windows 10 oder Windows Server 2016 oder höher.
- Der Hyper-V-Gast ist Windows 10 oder Windows Server 2016 oder höher.
- Das Gastbetriebssystem muss laufen.
- In dem Gastbetriebssystem muss der „Hyper-V VM Session Service“ (vmicvmsession) laufen.
- Der Aufrufer muss Hyper-V-Administrator sein.
- Die PowerShell-Konsole muss mit Administratorrechten laufen.
- Der Aufrufer braucht ein Benutzerkonto auf dem Gastbetriebssystem. Dieses Konto muss aber keine Administratorrechte haben!

Die Commandlets `New-PSSession` und `Invoke-Command` haben dafür die neuen Parameter `-VMName` und `-VMGUID` erhalten. Auf eine der beiden Weisen identifiziert der Nutzer die anzusprechende virtuelle Maschine.

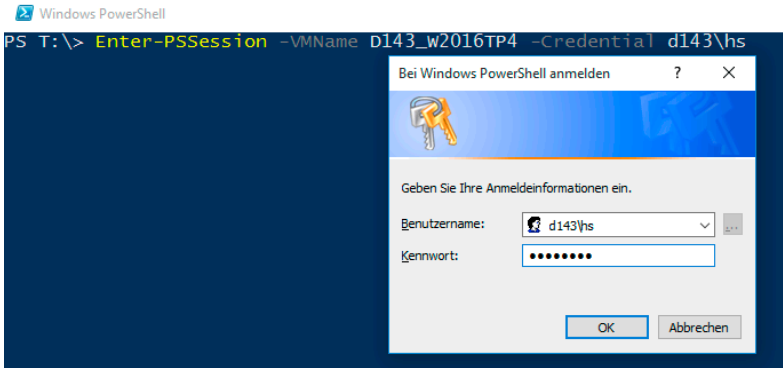


Bild 12.22 Initiieren einer Sitzung mit PowerShell Direct

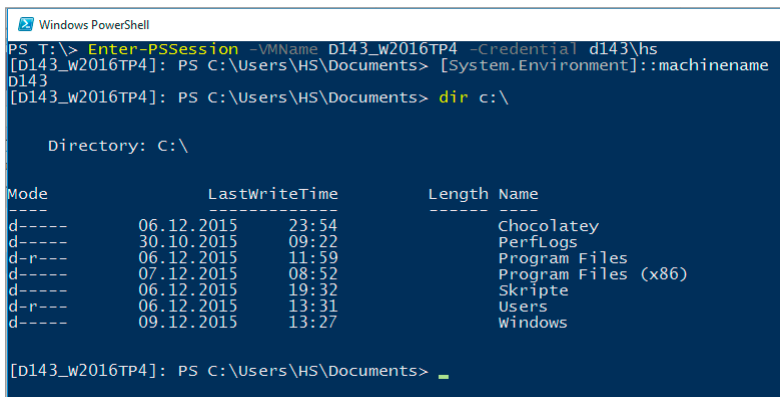


Bild 12.23 Nutzen einer PowerShell-Direct-Sitzung

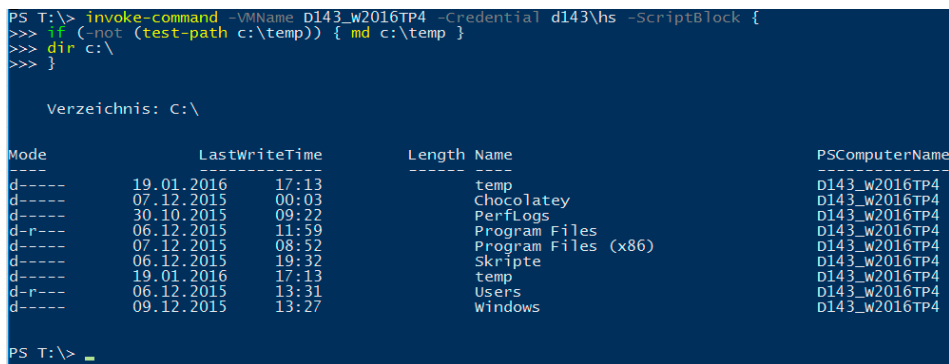


Bild 12.24 Ausführung eines Skriptblocks via PowerShell Direct





Die Befehle, die PowerShell Direct verwenden, müssen direkt auf dem Hyper-V-Host ausgeführt werden. Es ist nicht möglich, sich mit PowerShell Remoting mit dem Hyper-V-Host zu verbinden und diesem dann den Auftrag zu einer PowerShell-Direct-Verbindung zu geben. Diesen Versuch quittiert die PowerShell mit „Specified method is not supported.“



**TIPP:** Auch bei PowerShell Direct ist es möglich, das Kennwort für das Gastbetriebssystem im Quellcode eines Skripts zu verankern – wenn man sich der Risiken bewusst ist.

```
$VMName = "D130"
$cred = new-object System.Management.Automation.PSCredential -ArgumentList
"D130\hs", (ConvertTo-SecureString "geheim+123" -AsPlainText -force)
Enter-PSSession -vmname D130 -Credential $cred
```

## ■ 12.16 Praxisbeispiel zu PowerShell Direct

Das folgende Praxissskript zeigt, wie man eine frisch installierte virtuelle Maschine vom Hyper-V-Host aus konfiguriert:

- Festlegen einer statischen IP-Adresse
- Setzen des DNS-Servers
- Umbenennen des Computers
- Beitritt zu einer Domäne
- Neustart
- Alle PowerShell-Skripte erlauben
- PowerShell Remoting aktivieren
- Zugriff per RDP aktivieren

Danach ist dann eine Fernadministration des Rechners als neues Domänenmitglied per RDP und PowerShell Remoting möglich!

### Listing 12.6 [2\_Aufbauwissen\Remoting\WPS5\_PowerShell\_Direct\_PCKonfig.ps1]

```
# Konfiguration einer Windows-VM in HyperV
# (C) Holger Schwichtenberg, www.IT-Visions.de, 2013-2016
# Die folgenden Eingabedaten müssen vor dem Start angepasst werden!

# Zugang zur VM
$VMName = "F131"
# Kennwörter hier nur zu Testzwecken im Skript. Müssen ggf. interaktiv abgefragt
werden!
$cred = new-object System.Management.Automation.PSCredential -ArgumentList "$VMName\
administrator", (ConvertTo-SecureString "fbi+123" -AsPlainText -force)
```

```

$credDomain = new-object System.Management.Automation.PSCredential -ArgumentList
"fbi\administrator", (ConvertTo-SecureString ,fbi+123' -AsPlainText -force)

# Daten für neue Konfiguration
$ip = "192.168.1.131"
$DNSServer = "192.168.1.111"
$Gateway = "192.168.1.253"
$Domainname = "fbi.local"

# Skript-Einstellungen
$errorActionPreference = "stop"

#####

# Hilfsfunktion, die auf Verbindung zum Gast-Betriebssystem wartet
function WaitFor-PSDirect([string]$VMName, $cred){
    Write-Output "Warte auf PowerShell Direct-Verbindung zu $VMName mit Benutzer
 $($cred.username)"
    while ((invoke-command -VMName $VMName -Credential $cred {"Test"} -ea
 SilentlyContinue) -ne "Test") {Sleep -Seconds 1}
    "$VMName ist verfügbar via PowerShell Direct!"
}

##### Block 1:
# Festlegen einer statischen IP-Adressen
# Setzen des DNS-Servers
# Umbenennen des Computers
# Beitritt zu einer Domäne
# Neustart

$scriptblock1 = {
param($Name,$IP,$credDomain,$Gateway,$DNSServer,$Domainname)

$InterfaceName = (Get-netadapter | select-object -first 1).name
"Bisheriger Name $($[System.Environment]::machinename) mit Netzadapter $netzadapter"
Get-NetIPAddress
"Neuer Name: $Name mit IP $ip Gateway $Gateway DNSServer $DNSServer"
Set-NetIPInterface -InterfaceAlias $InterfaceName -Dhcp Disabled
New-NetIPAddress -InterfaceAlias $InterfaceName -IPAddress $IP -DefaultGateway
$Gateway -PrefixLength 24 | out-null
Set-DnsClientServerAddress -InterfaceAlias $InterfaceName -ServerAddresses $DNSServer
ping $ip
"Rename Computer und Domain Join..."
Add-Computer -NewName $Name -Verbose -DomainName $Domainname -Credential $credDomain
"PowerShell-Direkt-Skript #1 ist fertig!"
}

# Block 1 ausführen
"Konfiguriere VM $vmname - Schritt 1"
Invoke-Command -VMName $VMName -Credential $cred -ScriptBlock $scriptblock1
-ArgumentList $vmname,$ip,$credDomain,$Gateway,$DNSServer,$Domainname
"Reboot"

# Neustart und dann warten, bis Betriebssystem wieder zugreifbar!
Restart-VM $VMName -Force
WaitFor-PSDirect $VMName $cred

#####

```

```
# Alle PowerShell-Skripte erlauben
# PowerShell Remoting aktivieren
# Zugriff per RDP aktivieren

$scriptblock2 = {
param()
"PowerShell-Skripte erlauben..."
Set-ExecutionPolicy Unrestricted
"PowerShell-Remoting erlauben..."
Enable-psremoting -force -SkipNetworkProfileCheck
"RDP erlauben..."
set-ItemProperty -Path ,HKLM:\System\CurrentControlSet\Control\Terminal Server'-name
"fDenyTSConnections" -Value 0
Enable-NetFirewallRule -DisplayGroup "Remote Desktop"
set-ItemProperty -Path ,HKLM:\System\CurrentControlSet\Control\Terminal Server\
WinStations\RDP-Tcp' -name "UserAuthentication" -Value 1

"PowerShell-Direkt-Skript #2 ist fertig!"
}

# Block 2 ausführen
"Konfiguriere VM $vmname - Schritt 2"
Invoke-Command -VMName $VMName -Credential $cred -ScriptBlock $scriptblock2
"FERTIG: Fernadministration des Rechners $vmname als neues Domänenmitglied per RDP und
PowerShell Remoting nun möglich!"
```

# Stichwortverzeichnis

## Symbole

- & 51, 145
- > 188
- 32-Bit 10, 211
- 64-Bit 211
- \$\_ 67, 75, 123, 346
- \$args 257
- \$Args 107, 123
- \$Error 123, 165
- \$ErrorActionPreference 164, 787
- \$false 36, 40, 123
- \$global 121f., 865
- \$Home 123
- \$host 123, 215, 228, 504
- \$Input 123
- \$LastExitCode 123
- \$MaximumErrorCount 123
- \$null 97, 107, 151, 154, 187
- \$OFS 123
- \$PSHome 123
- \$PSHost 123
- \$psiSE 215
- \$PSItem 66, 346
- \$script 121
- \$StackTrace 123
- \$this 346, 865
- \$true 123
- \$window 865
- Bit 10
- expression 514
- force 40
- ItemsSource 868
- .NET 3, 30, 66, 79, 123, 141, 192, 278, 855, 887, 913, 927, 959
  - Klasse 275, 963
  - Runtime Host 29
- .NET Data Provider 581f.
- .NET Framework 29, 247, 379, 459, 662, 959f., 963
- 4.0 10

- .ps1;ps1 47
- [Type] 283

## A

- Ablaufverfolgung 3, 356
- About 115
- Absent 443
- AbsoluteTimerInstruction 311
- abstract 956
- Accelerator 119
- AccessControl 719, 724
- Access Control Entry 720
- Access Control List 720, 731
- Access Control Type 720
- Access Mask 720
- AccessMask 544
- AccountDisabled 756
- Account Manager 500
- AceFlags 720
- ACL 722, 735
- Active Directory 3, 197, 298, 433, 479f., 496, 746, 752, 756, 771, 795f., 876
  - PowerShell 771
  - Struktur 796
  - Suche 761
- Active Directory Application Mode 796
- Active Directory Domain Services 794
- Active Directory Service Interface *siehe* ADSI 737
- ActiveScriptEventConsumer 312
- Active-Scripting 109
- ActiveX Data Objects 581, 592, 742
- ADAccount 777
- ADComputer 777
- Add-ADGroupMember 781, 793
- AddCommand() 937
- Add-Computer 641f.
- Add-Content 528, 563, 579
- Add-DirectoryEntry 495
- Add-DistributionGroupMember 846
- Add-Feature 559
- Add-JobTrigger 374
- Add-LDAPObject 769, 878
- Add-Member 88, 343, 346
- Add-ODBCDSN 627
- Add-PSSnapin 890, 895
- AddScript 934
- AddScript() 937
- ADDSDeployment 794ff.
- Add-Type 287, 347, 530, 611
- Add-VirtualHardDisk 498
- Add-VMDisk 831
- Add-VMDrive 831
- Add-VMHardDiskDrive 809, 821
- Add-VMNIC 831
- Add-VMSwitch 809
- Add-WBSystemState 557
- Add-WindowsFeature 557, 660, 665ff., 794
- Administrator 270
- ADODB.Connection 743
- ADO.NET 581, 589, 742, 761
- ADPowerShell 771, 777
- AD RMS 480
- ADSI 739, 743f., 747, 749
  - Bindung 745f.
  - COM 743, 749
  - Container 751
  - .NET 737, 739, 743
  - Pfad 745
- AdsPath 761
- ADUser 777
- ADWS 772
- AgentPC 739
- Aktivierung 501
- Aktivität 386, 390

Alias 33, 42, 195, 519  
 Aliaseigenschaft 82, 87  
 AliasInfo 42  
 AllNodes 440  
 AllowEmptyCollection 882  
 AllowEmptyString 882  
 AllowNull 882  
 AllSigned 109  
 Alvin Kersh 738  
 -and 91  
 Änderungshistorie 586  
 Animation 868  
 Ankerelement 135  
 Anwendungspool 841, 843  
 Anzeigesprache 426  
 AppDomain 611  
 AppendChild(). 575  
 AppLockerPolicy 672f.  
 appSettings 226  
 Architektur 29  
 Array 141, 143, 145, 966  
 Aruk Kumaravel 744  
 AsJob 364, 383  
 ASP.NET 298  
 Assembly 286, 459, 465, 895, 924  
 - verbreiten 964  
 AssocClass 683  
 ASSOCIATORS OF 313  
 Assoziation 305  
 - WMI 302, 305  
 Attribut 951, 955, 965  
 - indiziert 966  
 Audio 282  
 Aufgabe  
 - geplant 369  
 Aufzählung 289  
 Ausdruck  
 - Regulär 133  
 Ausdruckauflösung 128  
 Ausdrucksmodus 49  
 Ausführungsrichtlinie 109  
 Ausgabeobjekt 898  
 Auslagerungsdatei 501  
 Authentifizierung 193, 756, 793  
 AuthorizationRuleCollection 725f.

## B

BackgroundColor 123, 215, 228  
 Background Intelligent Transfer Service 709  
 Backspace 129  
 Backup 557f., 610  
 Backup-GPO 801  
 Backup-SqlDatabase 608, 610  
 Base 761, 783

Basisauthentifizierung 245  
 Basisklasse 585  
 Batterie 647  
 Bedingung 152  
 Beep 129  
 Beep() 283  
 Befehl  
 - Extern 33, 50  
 Befehls-Add-On 58  
 Befehlsgeschichte 503  
 Befehlsmodus 49  
 Befehlsobjekt 590  
 Begin 532  
 BeginProcessing() 889, 893  
 Benutzer 297, 766, 952, 954  
 - anlegen 755  
 - löschen 757  
 - umbenennen 757  
 - verschieben 758  
 Benutzerabmeldung 512  
 Benutzeranmeldung 512  
 Benutzerdaten lesen 786  
 Benutzer-DSN 629  
 Benutzergruppe 792  
 Benutzerkennwort 756  
 Benutzerkontensteuerung 19, 106, 208  
 Benutzerkonto 786, 970  
 Benutzername 193  
 Benutzerschnittstelle 296  
 Berechnung 96  
 Best Practice 480, 853  
 Bezeichner 964  
 Beziehung 954  
 Bibliothek 962  
 Big Endian 761  
 Bild 494  
 Bildschirmschoner 349, 739  
 Binärdatei 579  
 Binärmodul 939  
 Bindung  
 - ADSI 745  
 - serverlos 746  
 - WMI 320  
 Bing 863  
 BIOS 297  
 Bitlocker 559  
 BitLocker  
 - Überblick 559  
 Bitmap 530  
 BITS 34  
 Blatt 743  
 BMC 295  
 Boot-Konfiguration 297  
 break 146, 148, 161  
 Bypass 110  
 ByPropertyName 69f.

Byte 126  
 ByVal 69  
 BZIP2 542

## C

C# 115, 347, 349, 873, 887f., 893, 959  
 C++ 962  
 CAB 60  
 Canvas 865  
 Carriage Return 129  
 CATID 314  
 C++/CLI 887  
 CD 285  
 ChangeAccess 551  
 Checkpoint-Computer 643  
 Checkpoint-VM 809, 824  
 Children 742  
 ChildSession 262  
 Chkdsk() 337  
 CHKDSK 297  
 Chocolatey 659  
 Chocolatey.org 658  
 Chrome 433, 658  
 CIM 295, 298  
 - Repository 306  
 CimClass 321, 329  
 CimClassProperties 329  
 CIM Explorer 222  
 CimInstance 321, 329  
 CimInstanceProperties 329  
 CimProperty 329  
 CIM Query Language *siehe* CQL  
 Cisco 295  
 City 778f.  
 class 167  
 ClassCreationEvent 412  
 ClassDeletionEvent 412  
 ClassModificationEvent 412  
 Clear-BitLockerAutoUnlock-Funktion 560  
 Clear-Content 564  
 Clear-DnsClientCache 694  
 Clear-Eventlog 713  
 Clear-EventLog 246  
 Clear-History 503  
 Clear-Host 159, 505  
 Clear-Item 519  
 Click 866  
 Clipboard 171  
 CLIXML 575  
 Close() 865  
 -cmatch 133  
 cmd.exe 66  
 cmdlet 921  
 Cmdlet Help Editor 924

- cn 753
- cnotmatch 133
- Codeausschnitt 212
- Codeeigenschaft 82, 87
- Color 649
- COM 30, 291, 529, 962
  - Kategorie 314
  - Komponente 297
  - Moniker 745
  - Sicherheit 311
- Commandlet 3, 33, 50, 53, 67, 201
  - erstellen 873, 881, 887
  - Klasse 889
  - Konvention 911, 928
  - Provider 196
  - Verkettung 909
- Command Line Event Consumer 313
- Command Mode 49
- CommandNotFoundException 165
- Comma-Separated Values 567
- CommitChanges() 281, 740, 749 f., 756
- Common Information Model *siehe* CIM
- Common Intermediate Language 960
- Common Language Runtime 960
- Common Language Specification 961
- Common Management
  - Information Protocol 296
- Common Parameter 39
- Common Type System 961
- compare 100
- Compare-Object 99 f.
- Compare-VM 809, 825 f.
- Complete-Transaction 359 f., 362
- Component Object Model *siehe* COM
- Compress-Archive 541
- Computer 304, 766, 954
- Computerguppe 226
- Computername 641
- ComputerName 107, 649
- Computerverwaltung 639
- configuration 434
- ConfigurationData 441
- ConfigurationID 451
- ConfigurationNamingContext 775
- confirm 40, 650, 689, 783, 786
- Confirm 39, 921
- Connection 588
- Connect-VMNetworkAdapter 809

- ConsolePaneBackgroundColor 215
- Console.WriteLine() 916, 929
- Container 751, 801
- Container-Klasse 743
- Continue 146, 148, 161, 163
- ConvertFrom-String 568
- ConvertFrom-StringData 424
- Convert-HTML 579
- Convert-String 567
- ConvertTo-CSV 567
- ConvertTo-DataTemplate 868
- ConvertTo-SecureString 793
- ConvertTo-WebApplication 842
- ConvertTo-XML 577
- Convert-VHD 809, 820, 823
- Convert-Xml 577
- copy 527
- Copy-GPO 800
- Copy-Item 161, 519, 527, 634
- Copy-NetFirewallRule-Funktion 697
- Copy/Paste 205
- Copy-ToZip 542
- Copy-VMFile 827
- CORBA 962
- Count 72, 143, 157
- Country 779
- CPU 100
- CQL 313, 327
- Create() 338
- CreateCommand() 590
- CreateElement() 575
- CreateInstance() 648
- CreationTime 530, 956
- Credential 793
- Credentials 781
- CSV 47, 191, 564 ff., 637, 789, 838
- CSV-Datei 101
- CultureInfo 898
- Cursor 585 f.
- CustomerID 883

## D

- Dana Scully 738
- DataReader 584 ff., 590, 592 ff.
- DataRow 87 f.
- DataSet 584 f., 592, 594 ff.
- Data Source Name *siehe* DSN
- DataTable 594 f.
- Datei 171, 297, 952
  - Eigenschaft 529, 534
  - kopieren 527
  - löschen 35
  - Rechte 297
  - verschieben 527
- Dateiname 33
- Dateinamenerweiterung 101, 103
- Dateisystem 3, 528, 723, 955, 965
- Dateisystemfreigabe 433, 544
- Dateiversionsverlauf 556
- Datenabfrage 314
- Datenbank 306, 495, 581
- Datenbankmanagementsystem 588
- Datenbankverbindung 588
- Datenbankzeile 87
- Datenbankzugriff 581
- Datenbereich 423
- Datenbindung 868
- Datendatei 423
- Datenmenge 195
- Datenquelle 627 f.
- Datenquellensteuerelement 584
- Datentyp 118, 123, 143, 747, 909, 966
  - .NET 67
  - PowerShell 118
  - WMI 301
- Datenzugriff 589
- DateTime 74, 77 f., 283
- Datum 140
- Day 74
- DB2 583, 627
- dBase 627
- DbCommand 590
- DbDataReader 592
- DbProviderFactories 583
- DCOM 306, 321, 904, 917
  - Konfiguration 297
- dcpromo 794
- DDL 313
- debug 916
- Debug 40
- Debugging 3, 393
- Debug-Modus 356
- DebugPreference 916
- Decimal 126
- Deep Throat 738
- Default Domain Policy 805
- DefaultNamingContext 775
- Deinstallation 655
- Delete() 956
- Deleting 649
- Delimiter 564
- Deployment 964
- Description 753, 779, 884
- Deserialisierung 254
- Desired State Configuration 429, 435

Desktop 297  
 Desktop-Anwendungen 961  
 Destruktor 952, 966  
 Dezimalzahl 125  
 DHCP 685 f., 688  
 Dialogfenster 192  
 diff 100  
 Digest 245  
 Directory 722 f.  
 DirectoryEntry 87 f., 278, 741 ff.,  
 747 ff., 752, 755, 758  
 DirectoryInfo 79, 91, 343, 534,  
 723, 956  
 Directory Management Objects  
 796  
 DirectorySearcher 762  
 DirectorySecurity 724  
 DirectoryString 753  
 Disable-ComputerRestore 643  
 Disable-JobTrigger 374  
 Disable-Mailbox 846  
 Disable-NetFirewallRule-Funktion  
 697  
 Disable-PnpDevice 647  
 Disable-PSRemoting 251  
 Disable-PSSessionConfiguration  
 261  
 Disable-VMIntegrationService  
 812  
 Disk Quotas 297  
 Dismount-VHD 820  
 DisplayName 779  
 Distinguished Name 745, 750,  
 753, 775, 779  
 Distributed COM 245  
 Distributed File System 297  
 Distributed Managements  
 Objects *siehe* DMO  
 DML 313  
 DMO 611  
 DMTF 295  
 DNS 272, 693, 794  
 DnsClient 687, 694  
 DNS-Client 690  
 DNSClient 688, 690  
 DNS-Konfigurationseinstellungen  
 – Per WMI abfragen 691  
 DNS-Server 298, 688, 692  
 do 146  
 DockPanel 865  
 Dokument 563  
 Dokumentation  
 – Active Directory 755  
 – .NET 62  
 Dollarzeichen 96, 128  
 Domain 697  
 Domain Controller 795

Domäne 641, 746, 797, 952,  
 954  
 – Beitritt 272, 641  
 – hinzufügen 641  
 DotNetTypes.Format.ps1xml 171,  
 175 f.  
 Dot Sourcing 47, 108 f., 459, 511,  
 874, 878  
 Double 126  
 DownloadString() 280, 705  
 DrivInfo 283  
 Driver 630  
 DriveType 288  
 Druckauftrag 648  
 – löschen 648  
 Drucker 171, 187, 297, 649  
 – verwalten 647 f.  
 Druckerport 648  
 Druckerverwaltung 647 f., 685  
 DSC 9, 429  
 – Linux 9  
 DSC Pull Server 446, 451  
 DSC *siehe* Desired State  
 Configuration  
 DSN 627, 629 f.  
 DuplexingMode 649  
 Durch 34  
 DVD 285, 826, 830

## E

echo 49  
 Eigenschaft 82  
 Eigenschaftenzwischenspeicher  
 740  
 Eigenschaftssatz 82, 84  
 Eingabe 191  
 Eingabeaufforderung 505 f.  
 Eingabemaske 855  
 Eingabeobjekt 906  
 Eingabesteuerelement 868  
 Einzelschrittmodus 354  
 Else 152  
 E-Mail 704, 707  
 – EmailEvent 316  
 EmailAddress 779  
 EmailEvent 316  
 Enable-BitLocker 561  
 Enable-ComputerRestore 643  
 Enable-JobTrigger 374  
 Enable-NetFirewallRule 703  
 Enable-NetFirewallRule-Funktion  
 697  
 Enable-ODBCPerfCounter 627  
 Enable-PnpDevice 647  
 Enable-PSRemoting 211, 249,  
 649, 702  
 Enable-PSSessionConfiguration  
 261, 263  
 Enable-VMIntegrationService 812  
 Encoding 564  
 End 532  
 EndProcessing() 889, 893  
 Enter-PSSession 251 f., 261, 265,  
 356  
 Enum 289  
 EnumerateCollection 900  
 Enumerationsklasse 288  
 Environment 431  
 Ereignis 951, 966  
 – PowerShell 411, 420  
 – WMI 311, 411  
 Ereignisabfrage 315  
 Ereigniskonsument 311 ff.  
 – permanent 311  
 – temporär 311  
 Ereignisprotokoll 101, 297, 307,  
 313, 433, 711  
 – Überwachung 316, 413  
 Ereignisprovider 311  
 Ereignissystem 411  
 Error 649  
 ErrorAction 40, 163 f., 529  
 ErrorBackgroundColor 21  
 ErrorRecord 160, 166  
 ErrorVariable 40  
 Ethernet 687  
 Event 299  
 EventConsumer 299  
 Event *siehe* Ereignis  
 EventViewerConsumer 312  
 Example 884  
 Exception 160, 165 f., 912, 918  
 Exchange Management  
 Shell 497, 845  
 Exchange Server 298, 307, 845  
 – Version 2007 62  
 ExecuteNonQuery() 590  
 ExecuteReader() 590, 592  
 ExecuteRow() 590  
 ExecuteScalar() 590  
 ExecutionPolicy 19 f.  
 EXIF 530  
 Exists() 748  
 exit 146  
 Exit-PSSession 253, 261  
 Expand-Archive 541  
 Export-Alias 47  
 Export-CliXml 575  
 Export-CLIXML 191  
 Export-Console 461, 896  
 Export-Counter 717  
 Export-Csv 91  
 Export-CSV 191, 567

Export-ModuleMember 465  
Export-VM 809, 825  
Export-VMSnapshot 824  
Express 604  
Expression 184  
Expression Mode 49  
Extended Reflection 67  
Extended Type System 67, 82,  
87, 329, 744  
Extensible Application Markup  
Language *siehe* XAML  
Extrinsic Event 311

## F

facsimileTelephoneNumber  
779  
Failover Cluster 480  
Fax 779  
FBI 738, 758  
Feature 662  
FeatureOperationResult 666  
Fehler 40  
Fehlerbehandlung 160, 962  
Fehlerklasse 146, 166  
Fehlermeldung 35  
Fehlersuche 353, 913  
Fehlertext 146  
Fernaufruf 246  
Fernausführung 107, 245  
– Hintergrundauftrag 367  
Fernverwaltung 245  
Fernzugriff 245  
Festplatte  
– virtuell 820  
Festplattenverschlüsselung 559  
Fibre-Channel 808  
Field 83  
File 722f., 953  
File History 556  
FileInfo 79, 91, 343, 534, 723,  
956  
FileInformation 675  
FileSecurity 723f.  
FileSystem 494, 519  
FileSystemAccessRule 726  
FileSystemInfo 956  
FileSystemObject 954  
FileSystemRights 289  
FileSystemWatcher 419  
FileVersionInfo 659  
filter 525  
Find() 742  
Find-Module 470f.  
Find-Package 658  
Firefox 433  
Firewall 500

Firewall-Regel 700  
First 73  
For 147, 150  
force 40  
Force 39, 783  
foreach 391  
ForEach 76, 100f., 146, 150, 742,  
902  
Foreach-Object 75f., 84, 96, 100,  
143, 150, 156, 355, 390, 563,  
901  
Forest 797  
Format 184  
Formatkennzeichner 180  
Format-List 65, 172, 724  
Format-Table 74, 84, 101, 172,  
177f., 184, 724  
Format-Wide 171f.  
Format-Xml 572f.  
Form Feed 129  
Fox Mulder 738  
FoxPro 627  
Framework Class Library 275  
Freigabe 553  
FullAccess 551  
FullArmor 479  
FullName 956  
function 33, 146, 154, 159, 195  
Function Prompt 505  
Funktion 33, 154, 195  
– eingebaut 159

## G

GAC *siehe* Global Assembly  
Cache  
Gast 807  
Gateway 688  
GeneralizedTime 753  
Geplante Aufgabe 369  
Gesamtstruktur 797  
Geschäftsanwendung 927  
GetAccessRules() 725f.  
Get-Acl 719, 723f., 735  
Get-ADComputer 781  
Get-ADDomain 797  
Get-ADDomainController 797  
Get-ADForest 797  
Get-ADGroup 781, 792  
Get-ADGroupMember 781, 793  
Get-ADObject 36, 737, 768, 781ff.  
Get-ADOptionalFeature 797  
Get-ADOrganizationalUnit 781,  
785  
Get-ADPrincipalGroupMember-  
ship 793  
Get-ADRootDSE 797  
Get-ADUser 781, 786f.  
Get-Alias 42  
Get-AppLockerFileInformation  
672  
Get-AppLockerPolicy 672f.  
Get-BitLockerVolume 560ff.  
Get-BPAModel 853  
Get-Bparesult 853  
Get-BPAResult 853  
Get-CDRomdrive 645  
Get-CDRomDrive 495  
Get-Childitem 541  
Get-ChildItem 34ff., 65f., 91, 94,  
101, 195, 525, 633, 653  
– BitLocker 561  
Get-ChildItem, 633  
Get-CimAssociatedInstance 321  
Get-CimClass 295, 321, 328  
Get-CimInstance 295, 316,  
321ff., 338, 649  
Get-Clipboard 859  
Get-Command 41, 53ff., 475  
Get-ComputerInfo 898  
Get-Computername 892  
Get-ComputerRestorePoint 643  
Get-Content 519, 529, 563, 579,  
637, 652  
Get-Counter 246, 715f.  
Get-Credential 57, 193, 268, 678,  
704  
Get-DataRow 495, 600  
Get-DataTable 495, 599  
Get-Date 77, 140f.  
Get-DhcpServer 22  
Get-DHCPServer 686  
Get-DirectoryChildren 495  
Get-DirectoryEntry 128, 495, 911  
Get-DirSize 873  
Get-Disk 520, 522, 645, 900f.,  
904, 906, 908f., 917  
Get-DisplaySetting 647  
Get-DnsClient 691  
Get-DnsClientCache 694  
Get-DnsClient-Funktion 690  
– Beispiel 691  
Get-DnsClientServerAddress 691  
Get-DnsClientServerAddress-  
Funktion 690  
Get-DomainController 22f., 746  
GetDrives() 282  
Get-DSCConfiguration 445  
Get-DVDDrive 498  
Get-Event 415  
Get-EventLog 16, 101, 246, 711f.  
Get-ExCommand 845  
Get-ExecutionPolicy 19  
Get-ExportedType 535



- GetFactoryClasses() 583
- Get-FileHash 531
- Get-FileVersionInfo 535, 659
- Get-FirewallRule-Funktion
  - Windows-Firewall 698
- Get-FloppyDrive 498
- Get-Flug 927
- Get-Flugziele 927
- Get-Font 641
- Get-FreeDiskSpace 522, 524
- Get-GPInheritance 804
- Get-GPO 800 f.
- Get-GPOReport 803
- Get-GPPermissions 806
- Get-GPPrefRegistryValue 805
- Get-GPRegistryValue 806
- Get-GPResultantSetOfPolicy 803
- Get-GPStarterGPO 800
- Get-Help 56 f., 60, 115, 873, 886, 923
- Get-History 503
- Get-Host 504
- Get-HotFix 246
- Get-Item 529, 633, 836, 841
- Get-ItemProperty 529, 633
- Get-Job 363, 365
- Get-JobTrigger 374
- Get-Keyboard 646
- Get-LDAPChildren 769, 878
- Get-LDAPObject 769, 878
- Get-Location 41, 519
- Get-LogicalDiskInventory 520
- GetLongDateString() 77
- GetLongTimeString() 77
- Get-Mailbox 845 f.
- Get-Mailboxdatabase 845
- Get-Member 78, 80, 82 f., 92, 100, 287, 329, 744, 901
- Get-Members 758
- Get-MemoryDevice 495, 645
- Get-Methode 83
- Get-Module 465, 473, 939
- Get-MountPoint 22
- Get-MultiTouchMaximum 647
- GetNames() 289
- Get-NetAdapter 687
- Get-NetAdapterBinding 687
- Get-NetFirewallAddressFilter-Funktion 697
  - Windows-Firewall 699
- Get-NetFirewallApplicationFilter-Funktion 697
- Get-NetFirewallInterfaceFilter-Funktion 697
- Get-NetFirewallInterfaceType-Filter-Funktion 697
- Get-NetFirewallPortFilter-Funktion 697
- Get-NetfirewallProfile 698
- Get-NetFirewallProfile-Funktion 697
- Get-NetFirewallRule-Funktion 697
- Get-NetIPInterface 688
- Get-NetworkAdapter 495, 646
- GetObject() 293
- Get-ODBCDriver 627
- Get-ODBCDSN 627
- Get-OSVersion 639
- Get-Package 658 f.
- Get-PackageProvider 657
- Get-PackageSource 658
- Get-Passagier 927
- Get-PipelineInfo 78 f., 92, 910
- Get-PnpDevice 647
- Get-PnpDeviceProperty 647
- Get-PointingDevice 646
- Get-PowerShellDataSource 868
- Get-Printer 649
- Get-PrintJob 649
- Get-Process 33 f., 36 f., 41 f., 47, 65, 70, 75 ff., 80, 82, 91 f., 96, 100, 178, 187, 246, 355, 677, 909, 911
- Get-Processor 495, 645 f.
- Get-PSBreakpoint 358
- Get-PSDrive 520
- Get-PSProvider 197
- Get-PSRepository 471
- Get-PSSession 261
- Get-psessionconfiguration 262
- Get-PSSessionConfiguration 261
- Get-PSSnapIn 462
- Get-PswaAuthorizationRule 226
- Get-Random 126
- GetRelated() 682
- Get-ReparsePoint 540
- Get-Service 70, 72, 77, 91, 107, 187, 246, 681
- Get-service | where status -eq stopped | Out-Printer -Name 187
- Get-SHA1 532
- GetShortDateString() 77
- Get-ShortPath 526
- GetShortTimeString() 77
- Get-SmbShare 545, 551
- Get-SmbShareAccess 552
- Get-SoundDevice 645
- Get-SqlData 616
- Get-Storagegroup 845
- Get-Tapedrive 645
- Get-TargetResource 458
- GetTempName() 292
- Getter 83, 966
- Get-TerminalSession 22
- Get-TraceSource 356
- Get-Transaction 359, 362
- GetType() 78, 123, 185
- Get-USBController 495, 646
- Get-Variable 118, 124
- Get-VHD 820
- Get-Videocontroller 495, 645
- Get-VirtualHardDisk 498
- Get-VM 809 f., 820
- Get-VM BIOS-VM 811
- Get-VMBuildScript 829, 831
- Get-VMHost 498, 809
- Get-VMMemory 811
- Get-VMProcessor 811
- Get-VMSnapshot 824
- Get-VMSummary 831
- Get-VMThumbnail 831
- Get-VMThumbnail 830
- Get-WBPolicy 557
- Get-WBSummary 558
- Get-WebApplication 836
- Get-WebitemState 843
- Get-Website 836, 841
- Get-WebvirtualDirectory 836
- Get-WindowsEdition 640
- Get-WindowsFeature 660, 662 ff.
- Get-WinEvent 246
- Get-WmiObject 30, 246, 285, 295, 321 ff., 326, 524, 645 f., 651, 682 f., 685, 715, 739
- Get-Wmiobject Win32\_Processor 646
- Get-xDscOperation 452
- Gigabyte 126
- Gitternetz 865
- GivenName 778 f.
- Gleichheitszeichen 145
- Global 865
- Global Assembly Cache 286
- Global Unique Identifier 750, 963
- gm 100
- Google 863
- GPMC 799
- Grafikkarte 297, 322
- Grant-SmbShareAccess 552
- Grid 865 f.
- GridView 174
- Group 94, 431, 743
- GROUP BY 313, 412
- Group-Object 94, 100 f., 712
- Gruppe 766, 781, 789 f.
  - anlegen 759
  - auflisten 758
  - Mitglied aufnehmen 759
- Gruppenmitglieder 781

Gruppenmitgliedschaft 760  
Gruppenrichtlinie 479f., 512,  
799ff., 804  
– Vererbung 804  
Gruppierung 94  
GUID 524  
GZIP 542

## H

Hardlink 538  
Hardware 297, 495  
Hardwareverwaltung 645  
Hash-Tabelle 143f., 279, 338  
Hashtable 143, 145, 279, 440f.,  
458  
HAVING 313, 412  
Heimordner 123  
Help-Info 60  
HelpMessage 882  
Herausgeber 112  
Here-String 127  
Herunterfahren 642  
Hexadezimalzahl 125  
hidden 169  
Hilfe 53  
Hintergrundauftrag 363  
Hintergrundübertragungsdienst 34  
History 503  
HKCU 195  
HKEY\_CURRENT\_USER 634  
HKEY\_LOCAL\_MACHINE 634  
HKLM 195  
HomeDirectory 779  
HomeDrive 779  
Host 807  
Hosting 931  
Hotfix 297  
HTML 579, 840, 855  
HTTP 245, 446  
HTTPS 245  
Hyper-V 270, 499, 807, 827  
– Überblick 807  
Hyper-V-Integrationsdienste 812  
Hypervisor 807  
Hyper-V-Modul  
– Überblick 808

## I

IADs 740f., 743  
IADsComputer 743  
IADsContainer 741, 743  
IADsGroup 743  
IADsUser 743, 756  
idempotent 429f.

Identität 841  
Identity 720  
IdentityReference 728  
IDL 308  
IEnumerable 742  
if 146, 152  
IIS 197, 223, 746  
– 8.0 833  
– Internet Information Server  
833  
IIS-Anwendung 842  
ILSpy 287  
-imatch 133  
Impersonifizierung 193, 748  
– WMI 310  
Implizites Remoting 264  
Import-Alias 47  
Import-CliXml 503, 576  
Import-CLIXML 191  
Import-Counter 717  
Import-Csv 101, 526  
Import-CSV 191, 566, 789  
Import-GPO 801  
Import-INIFile 570  
Import-Module 393, 465, 475,  
944f.  
Import-PSession 265  
Import-VM 809, 825f.  
IncludeUserName 678  
Index 141  
Indexer 966  
Informix 583, 627  
Ingres 583  
Inheritance Flags 720  
INI-Datei 570  
inlinescript 388  
Innertext 575  
-inotmacht 133  
InputBox 287  
InputBox() 192  
-InputObject 70  
InputObject 287, 911, 913  
Inquire 163  
Install-ADDSDomainController  
795  
Install-ADDSForest 795  
Installation 654  
Installationsordner 10, 86, 123  
Installationstechnologie 654  
Install-Module 469, 471  
Install-PswaWebApplication  
225  
installutil.exe 890, 895  
Install-WindowsFeature 225,  
447  
InstanceCreationEvent 412f.,  
415

InstanceDeletionEvent 299, 311,  
412  
InstanceModificationEvent 311,  
412  
Instanz 279, 952  
Instanziierung 953  
Instanzmitglied 966  
int 118f.  
Int32 119  
Int64 126  
INTEGER 753  
Integrated Scripting Environment  
210, 931  
IntelliSense 24, 34, 240  
Interface 954  
Interface Definition Language  
308  
InternalHost 504  
International .NET Association  
XXIV  
Internet Control Message  
Protocol 695  
Internet Information Server 298,  
833  
Internet Information Services  
296, 298, 446, 453, 682, 746,  
833  
Interpretermodus 207  
IntervallTimerInstruction 311  
Intrinsic Event 311  
InvalidOperationException 742  
Invoke-BPAModel 853  
Invoke-CimMethod 321, 338  
Invoke-Command 251, 253, 255,  
257ff., 261, 267, 270, 356, 364  
Invoke-DbCommand 599  
Invoke-DBMaint 613  
Invoke-Expression 145  
Invoke-History 503  
InvokeMethod() 320  
Invoke-Query 616  
Invoke-SqlBackup 624  
Invoke-SqlCmd 602, 604, 609f.,  
621, 624  
Invoke-SQLCmd 608  
Invoke-SqlCommand 495  
Invoke-webRequest 706  
Invoke-WmiMethod 321, 337  
IP  
– Adresse 314, 685  
– Konfiguration 314  
IPAddress 688  
IP-Adresse 272, 433, 685, 688  
ipconfig 50  
IPHostEntry 693  
IP Routing 297  
IRQ 297

ISA 313  
 ISE  
 – Integrated Scripting Environment 210  
 IsePack 494  
 ISE *siehe* Integrated Scripting Environment  
 ISO 807, 814, 817, 826  
 Item() 741

## J

Java 962  
 Jeffrey Snover 105  
 Job  
 – zeitgesteuert 373  
 Job-Trigger 373  
 – Zeitgesteuerte Jobs 373  
 John Doggett 738  
 -Join 132  
 Join() 132  
 Join-String 132  
 JPEG 530  
 JScript .NET 347  
 JSON 707  
 Junction 539  
 Junction Point 538 f.  
 Just-In-Time-Compiler 960

## K

Kennwort 193, 739, 756  
 Kerberos 245, 311  
 Kill() 76  
 Kilobyte 126  
 Klammer 36  
 – rund 135  
 Klammeraffe 49  
 Klasse 118, 282, 300, 304, 952, 954 f., 965  
 – CIM 298  
 – COM 276, 291  
 – Commandlet 889, 908, 930  
 – .NET 167, 275, 889, 963 f.  
 – PowerShell 167  
 – statisch 283  
 – WMI 296, 298  
 Klassendiagramm 956 f.  
 Klassenhierarchie 956  
 Klassenmitglied 282, 966  
 Klassenname 278  
 Kommandomodus 207  
 Kommandozeilenbefehl 50  
 Kommentar 116  
 Komponentenorientierung 960, 962  
 Komposition 868

Komprimierung 541 ff.  
 Konstruktor 952, 966  
 Konstrukturfunktion 277 ff.  
 Konstrukturfunktion 277, 292  
 Kontakt 766  
 Konvention 928  
 Kopieren/Einfügen 205  
 Kreuzzuweisung 145

## L

Label 184  
 LastAccessTime 956  
 Laufwerk 195 f., 202, 520, 634  
 – virtuell 820  
 LDAP 744, 746, 761  
 – Suchanfrage 741, 761  
 – Suche 767  
 LDAP-Query 762  
 Leaf 751  
 Leistung 715  
 Leistungsindikator 716  
 Length 72  
 Length 72, 343  
 liefert Get-WBBackupSet 558  
 Limit-EventLog 246, 711, 713  
 LinearGradientBrush 866  
 Linie 955  
 Linux 9, 963  
 Literal 180  
 Little Endian 761  
 Lizenzierung 501  
 Log File Event Consumer 313  
 Logoff 512  
 Logon 512  
 Lokalisierung 304, 426  
 Loopback 248

## M

Machine.config 583  
 MachineName 247, 893  
 Mac OS 9, 963  
 MailMessage 703  
 MAML 60, 923  
 Manage-Bde 559 f.  
 Managed Code 740  
 Managed Object 296  
 Managed Object Format 308  
 Managed Provider 581, 761  
 ManagementBaseObject 319  
 ManagementClass 87 f., 319, 321, 324, 329, 900  
 ManagementEventWatcher 414  
 ManagementObject 88, 319, 320 f., 324, 329 f., 337, 682, 900 f.

ManagementObjectCollection 329, 900 f.  
 ManagementObjectSearcher 324, 900  
 ManagementScope 413  
 Mandatory 881, 904  
 Manifest 466  
 Manifestmodul 939  
 Maschinencode 960  
 -match 133  
 Match 88  
 MaximumDriveCount 202  
 maxSessionsAllowedPerUser 226  
 md 529, 531, 637  
 measure 96  
 Measure-Command 355  
 Measure-Object 96, 100  
 Measure-VM 809  
 Megabyte 126  
 Mehrsprachigkeit 426  
 Merge-VHD 820  
 Message 160  
 MessageBox 192  
 Metamodell 306  
 Metaobjekt 750  
 Methode 82, 951, 955, 966  
 – Getter 966  
 – Setter 966  
 Method *siehe* Methode  
 Microsoft Access 596  
 – Treiber 587  
 Microsoft-Access 11  
 Microsoft Access Driver 630  
 Microsoft.ACE.OLEDB 587  
 Microsoft Certified Solution Developer XXIII  
 Microsoft Developer Network 62, 275  
 Microsoft Excel 789  
 Microsoft Exchange 795  
 Microsoft Exchange Server 433, 497, 845  
 Microsoft.GroupPolicy 800  
 Microsoft.Jet.OLEDB 587  
 Microsoft.Management.Infrastructure.CimClass 329  
 Microsoft.Management.Infrastructure.CimClassProperties 329  
 Microsoft.Management.Infrastructure.CimInstance 329  
 Microsoft.Management.Infrastructure.CimInstanceProperties 329  
 Microsoft.Management.Infrastructure.CimProperty 329  
 Microsoft Office 298

- Microsoft Outlook 680
  - Microsoft Print Ticket XML 649
  - Microsoft SQL Server 582, 602
  - Microsoft.Vhd.PowerShell 820
  - Microsoft.VisualStudio.Interaction 192
  - Microsoft.Win32 653
  - Microsoft.Win32.RegistryKey 633
  - Microsoft Word 293
  - Minute 74
  - Mitglied 951
    - .NET 965
    - statisch 966
    - WMI 333
  - MMC 310
  - Modul 465, 472
  - Module Browser 469
  - Modulo 145
  - MOF 308, 439, 441
  - Monad 4
  - Monica Reyes 738
  - Moniker 745
  - Mono 963
  - Month 74
  - more 159, 179
  - Most Valuable Professional XXIII
  - Mount-SpecialFolder 521
  - Mount-VHD 820
  - move 527
  - Move-ADObject 781f.
  - Move-Item 519, 527
  - Move-Mailbox 846
  - Move-VM 809
  - MSCL 29
  - mscorlib.dll 964
  - MSDN 861
  - MSDN Library 62, 275
  - MSFT\_Printer 649
  - MSFT\_PrintJob 649
  - MSFT\_SmbShare 551
  - MSFT\_SmbShareAccess-ControlEntry 552
  - MSI 454, 654f., 890
  - MTA 860
  - Multithreading 962
  - MySQL 593
  - MySQL 583, 593, 602, 613
  - MySQLConnection 594
  - MySQLLib 602
- N**
- Name 70, 779
  - Namensauflösung 693
  - Namensraum 301, 303ff., 722, 796, 963
    - ADSI 745
    - .NET 963f.
    - WMI 303, 305
  - Namensraumhierarchie 963
  - NamespaceCreationEvent 412
  - NamespaceDeletionEvent 412
  - Namespace-ID 745
  - NamespaceModificationEvent 412
  - Nano Server 9
  - NativeObject 742, 744
  - Navigation 195
  - Navigation Provider 196
  - Navigationsbefehl 198
  - Navigationsmodell 519
  - Navigationsparadigma 195
  - Navigationsprovider 737
  - NetAdapter 687f.
  - NetSecurity 696
  - NetSecurity-Modul
    - Überblick 696
  - Netsh 690, 692
  - Netsh
    - Per PowerShell aufrufen 692
  - netstat 50
  - NetTCPIP 687f.
  - NetworkInterface 898
  - Network Load Balancing 298
  - Netzlaufwerk 297
  - Netzlaufwerksverbindung 297
  - Netzwerkadapter 807
  - Netzwerkcenter 250
  - Netzwerkkarte 297, 314, 685, 952
    - Geschwindigkeit 689
  - Netzwerkkartenprofil 702
  - Netzwerkkonfiguration 501, 685
  - Netzwerkmanagement 295, 479
  - Netzwerkprofil
    - Per PowerShell setzen 702
  - Netzwerkverbindung 297, 433, 687f.
  - Neustart 272, 666
  - Neustarten 642
  - new() 277, 279, 292
  - New-ADGroup 781, 792
  - New-ADObject 782
  - New-ADOrganizationalUnit 781, 785
  - New-ADUser 781, 786, 788
  - New-AppLockerPolicy 672, 675
  - New-Buchung 927f.
  - New-Button 57, 864, 866
  - New-CheckBox 868
  - New-CimInstance 321, 338
  - New-CimSession 323
  - New-CimSessionOption 323
  - New-ComboBox 868
  - New-DSCChecksum 451
  - New-Ellipse 868
  - New-Event 420
  - New-EventLog 246, 711, 713
  - NewFirewallRule 700
  - New-GPLink 801
  - New-GPO 800
  - New-GPStarterGPO 800
  - New-Grid 866
  - New-Guid 277
  - New-Hardlink 539
  - New-HardwareProfile 498
  - New-Image 868
  - New-Int64Animation 868
  - New-Item 196, 519, 536, 564, 633f., 637
  - New-Itemproperty 635, 637
  - New-JobTrigger 374, 376
  - New-Junction 540f.
  - New-Label 861, 866
  - New-Line 129
  - New-Line 868
  - New-ListBox 868
  - New-Mailbox 846
  - New-Mailboxdatabase 846
  - New-MediaElement 868
  - New-Menu 868
  - New-Module 465
  - New-NetFirewallRule-Funktion 697
    - Windows-Firewall 700
  - New-NetIPAddress 688
  - New-Object 276ff., 284, 286, 291, 967
  - New-PasswordBox 866
  - New-ProgressBar 868
  - New-PSDrive 202, 634, 653, 776
  - New-PSSession 249, 251, 261, 264, 268, 270
  - New-RadioButton 868
  - New-Rectangle 868
  - NewRichTextBox 868
  - New-ScheduledJobOption 377
  - New-ScrollBar 868
  - New-SelfSignedCertificate 111
  - New-Service 681
  - New-Shortcut 537
  - New-Slider 868
  - New-SmbShare 550f.
  - New-StatusBar 868
  - New-Storagegroup 846
  - New-Storyboard 868
  - New-TextBlock 868
  - New-TextBox 866
  - New-TimeSpan 141
  - New-TreeView 868
  - New-UrlShortcut 538
  - New-VHD 816, 820f.
  - New-ViewBox 868

New-VirtualDVDDrive 498  
 New-VirtualNetworkAdapter  
   498  
 New-VM 498, 809, 814, 831  
 New-VMSwitch 810  
 New-WebApplication 842  
 New-WebAppPool 841  
 New-WebserviceProxy 708  
 New-WebServiceProxy 706 ff.  
 New-Website 837  
 New-WebVirtualDirectory 842  
 New-Window 866  
 New-Zip 542  
 NoAccess 551  
 Node 575  
 Non-Terminating Error 160, 917,  
   920  
 Northwind 604, 625  
 Notation 955  
   – umgekehrt polnische 761  
 NoteProperty 343, 570  
 Notes 884  
 Notizeigenschaft 82, 85, 92  
 -notmatch 133  
 Novell 746  
 Now 283  
 Nslookup 690, 694  
 NtAccount 727  
 NTAccount 724, 726, 728  
 NT Event Log Event Consumer  
   313  
 NTFS 307  
 NTLM 245  
 NTSecurityDescriptor 753  
 NuGet Package Manager 239

## O

Object 956  
 Object[] 157  
 ObjectCategory 753, 765, 784  
 ObjectClass 753, 765, 778 f.,  
   784  
 ObjectGUID 754, 779  
 Object Linking and Embedding  
   Database 592  
 ObjectSecurity 722 f.  
 ObjectSecurityDescriptor 753  
 ObjectSid 753  
 ObjectVersion 754  
 Objekt 568, 929, 951 f., 954 f.  
   – Dynamisch 343  
   – .NET 857  
   – WMI 298  
 Objektadapter 88, 329, 584  
 Objektassoziation  
   – WMI 305

Objektbaum 954 f.  
 Objektidentifikation  
   – ADSI 745 f.  
 Objektmenge 526  
 Objektorientierung 66, 951,  
   960 f.  
 Objekt-Pipeline 526  
 Objekttyp 953  
 OData 446  
 ODBC 581 f., 627, 630  
 Office 779  
 ogv 171  
 OLEDB 581 f., 592, 742  
   – Provider 742, 761  
 OleDbCommand 590, 596  
 OleDbConnection 588, 592,  
   596  
 OleDbDataAdapter 596  
 OLEDB-Provider 596  
 On\_Click 866  
 OneGet 656  
 OneLevel 783  
 ONELEVEL 761  
 Open() 588  
 Open Database Connectivity  
   – Einstellung 297  
 Operator 90, 132, 142, 145  
 Optimize-VHD 820  
   –or 91  
 Oracle 582, 593  
 OracleCommand 590  
 OracleConnection 588  
 Ordner 34, 51, 199, 525 ff.  
   – Dateisystem 297  
 Organisationseinheit 766, 789 f.  
   – anlegen 760  
 OutBuffer 40  
 Out-Clipboard 171  
 Out-Default 171, 175, 934, 936  
 Out-File 171, 188  
 Out-GridView 57, 171 f., 174 f.  
 Out-Host 171, 178 f.  
 Outlook 94, 500  
 Out-Null 171 f., 186  
 Out-Printer 171, 187, 648  
 Out-Speech 171, 189  
 Out-SqlScript 613  
 OutVariable 40, 98  
 ov *siehe* OutVariable

## P

PackageManagement 467  
 Page File 501  
 Panel 865  
 PaperSize 649  
 Papierkorb 529

parallel 390  
 -Parameter 57  
 Parameter 158, 884, 904, 913  
   – Abkürzung 37 f.  
   – Skript 107  
 Parameterliste 107  
 ParentSession 262  
 parsen 567  
 Partition 781  
 PascalCasing 964  
 PASH 9  
 PassThru 98, 781  
 Pause 649  
 Performance Counter Provider  
   715  
 Performance Monitor 297, 307  
 PERL 115  
 Persistenz 392  
 Petabyte 126  
 Pfad 304  
   – ADSI 745  
   – Verzeichnisdienst 745  
   – WMI 301, 303 f.  
 Pfadangabe 199  
 Pfeilspitze 956  
 Pflichtparameter 16  
 PHP 115  
 PhysicalDeliveryOfficeName 754,  
   779  
 PIN 560  
 Ping 50, 297, 695, 917  
 Ping-Host 22 ff., 695  
 Ping-VM 831  
 Pipe 66  
 Pipeline 3, 29, 65, 67, 78, 96, 186,  
   333  
   – Ausgabe 898  
   – Eingabe 906  
 Pipeline Processor 68, 890  
 PipelineVariable 40, 99, 178  
 Pipelining 65, 195  
 Plattform Invoke 349  
 Plattformunabhängigkeit 960  
 Platzhalter 180  
 Plug-and-Play 647  
 Polymorphismus 958  
 PoshConsole 233  
 Position 882, 904  
 Postfach 846  
 Postfix-Notation 761  
 PowerGadget 479  
 PowerGUI 210  
 Power Management 307  
 PowerShell 3, 42, 65  
   – 4.0 429  
   – Extension 495  
   – Hosting 3

- Konsole 203
- Laufwerk 195, 202, 634
- Remoting 241
- Sicherheit 109
- Skriptsprache 115
- PowerShell Analyzer 235
- PowerShell Community
  - Extensions 490, 737
- PowerShell Direct 270, 272
- PowerShell.exe 476
- PowerShell Gallery 431, 468
- PowerShellGet 467f.
- PowerShell Management Library
  - for Hyper-V 808, 828
- PowerShell Plus 911
- PowerShellPlus 210, 230, 241
- PowerShell Remoting 272, 649
  - Port 270
- PowerShell-Remoting 223
- PowerShell Script Analyzer 216
- PowerShell Web Access *siehe*
  - PSWA
- PowerStudio 240
- PowerTab 239
- PrimalScript 236, 241
- Principal 724
- Printer 171
- Printing 649
- PrintManagement 648
- Print Ticket XML 649
- Private 697
- Privileg 311
- Process 70, 92, 431, 532, 911
- ProcessRecord() 889, 893, 904
- Professional Developer
  - Conference 4
- ProfilePath 779
- profile.ps1 286, 874, 913
- Profilskript 209, 506
- Programmcodeanalyse 216
- Programmgruppe 297
- Programmiersprache 152
- Programmiersprachen-
  - unabhängigkeit 961
- Prompt 505
- Propagation Flags 720
- Property 83, 178, 966
- PropertyY 951
- Property Cache 749
- PropertyCollection 741
- PropertyDataCollection 320, 329
- PropertyGrid 857
- PropertyNames 741
- PropertyValueCollection 741, 748
- ProtectedFromAccidental-
  - Deletion 779, 783
- Protokolldatei 313

- Provider 198
  - ADO.NET 581
  - Dateisystem 519
  - PowerShell 196
  - Verzeichnisdienst 737
- WMI 298
- Proxy 707
- Prozedur 154
- Prozess 34, 100, 297, 306
  - auflisten 187, 677
  - beenden 680
- Prozessor 808
- PSBase 744, 750f.
- PSCmdlet 889
- PSCodeGen 494
- PSComputerName 259
- PSCredential 193, 678, 793
- PSCustomObject 92, 345, 565,
  - 570
- PSCX 23, 469, 536, 542f.
- psdl 425f.
- PSDiagnostics 480
- PSDriveInfo 520
- PSDSCRunAsCredential 434
- PSHost 933
- PSHostRawUserInterface 933,
  - 934
- PSHostUserInterface 933f.
- PSImageTools 494
- PSModulePath 466, 940
- PSObject 87, 936f.
- PSReadline 206
- PSRemotingJob 364
- PSRSS 494
- PSScheduledJob 369
- PSSession 260
- PSSnapIn 891
- PSSystemTools 494, 520, 646
- PSUserTools 494
- PSVariable 118
- PSWA 223, 225
- Public 697, 904
- Public Network 250
- Punktnotation 74, 280, 333
- Put() 334f.
- pv *siehe* PipelineVariable
- Python 115

## Q

- Quantifizierer 135
- Quantor 135
- QueryDialect 327
- Quest 479, 496

## R

- RawUI 215
- RDP *siehe* Remote Desktop
  - Protocol
- ReadAccess 551
- Read-Host 191, 385f.
- Receive-Job 363, 365f., 383
- Rechenleistung 100
- recurse 36, 525
- Recurse 781
- recursive 786
- Recursive 781
- Redirection *siehe* Umleitung
- REFERENCES OF 313
- Refresh() 956
- RefreshCache() 749
- RefreshFrequencyMins 451
- Regel 672, 700
- Register-CimIndicationEvent 321,
  - 419
- Register-DnsClient-Funktion 690
- Register-Event 417
- Register-ObjectEvent 555
- Register-Packagesource 470,
  - 658
- register-psSessionConfiguration
  - 263
- Register-PSSessionConfiguration
  - 261, 263
- Register-ScheduledJob 375f.
- Register-WmiEvent 419
- Register-WMIEvent 414
- Registrierungsdatenbank 3, 195,
  - 201f., 297, 443, 633
  - Schlüssel 633
- Registry 305, 307, 431, 633
- RegistryKey 653, 722f.
- RegistrySecurity 724
- RegistryValueChangeEvent 311,
  - 412
- Regulärer Ausdruck 133
- Relative Distinguished Name
  - 750f.
- Remote Desktop 702
- Remote Desktop Protocol 702
- Remote Desktop Service 480
- Remote Procedure Call 246
- Remote Server Administration
  - Tools 480, 737, 773
- Remoting 107
- Remove-ADGroup 793
- Remove-ADGroupMember 781,
  - 793
- Remove-ADObject 781ff.
- Remove-ADOrganizationalUnit
  - 785

Remove-ADUser 40, 786  
 Remove-Buchung 927  
 Remove-CimInstance 321, 339  
 Remove-Computer 642  
 Remove\_DirectoryEntry 921  
 Remove-DirectoryEntry 495, 911  
 Remove-Event 417  
 Remove-EventLog 246, 71  
 Remove-GPLink 801  
 Remove-GPO 800  
 Remove-GPPrefRegistryValue 806  
 Remove-GPRegistryValue 806  
 Remove-Item 35, 39 f., 519, 527, 634, 637  
 Remove-ItemProperty 636  
 Remove-Job 363, 366  
 Remove-JobTrigger 374  
 Remove-LDAPObject 769, 878  
 Remove-Module 465, 477  
 Remove-NetFirewallRule-Funktion 697  
 – Windows-Firewall 701  
 Remove-NetIPAddress 688  
 Remove-NetRoute 688  
 Remove-ODBCDsn 627  
 Remove-PrintJob 649 f.  
 Remove-PSBreakpoint 358  
 Remove-PSSession 261 f.  
 Remove-PswaAuthorization-Rule 226  
 Remove-SmbShare 39 f., 551  
 Remove-Variable 122  
 Remove-VM 809, 819 f.  
 Remove-VMSnapshot 824  
 Remove-WebApplication 844  
 Remove-WebAppPool 844  
 Remove-Website 844  
 Remove-WebVirtualDirectory 844  
 Remove-WindowsFeature 660, 666  
 Remove-WmiObject 321, 339  
 Rename-ADObject 781 f.  
 Rename-Computer 641  
 Rename-Drive 524  
 Rename-GPO 800  
 Rename-Item 527  
 Rename-NetAdapter 690  
 Rename-NetFirewallRule-Funktion 697  
 Rename-VM 809  
 Rename-VMSnapshot 824  
 Repair-VM 809  
 Replikation 297  
 Repository 306  
 Resize-VHD 810, 820  
 Resolve-DnsName 694

Resolve-DNSName-Funktion  
 – Beispiel 694  
 Resolve-DsnName-Funktion 690  
 Resolve-Host 693  
 Resolve-Path 199  
 ResponseHeaders 280  
 Ressource 431  
 REST 707  
 Restart-Computer 246, 642, 667  
 Restart-PrintJob 649  
 Restart-Service 267, 269, 681, 683 f.  
 Restart-VM 809  
 Restore-ADObject 782  
 Restore-Computer 643  
 Restore-DscConfiguration 443  
 Restore-GPO 801  
 Restore-VMSnapshot 824 f.  
 Restricted 109  
 Resume-PrintJob 649  
 Resume-Service 681, 683  
 Resume-VM 809  
 return 146, 201, 890  
 Revoke-SmbShareAccess 552  
 Richtlinienergebnisbericht 803  
 Rolle 662  
 Rollendienst 662  
 rootcimv2 305  
 RPC *siehe* Remote Procedure Call  
 RSAT 808  
 RSS 494, 705 f.  
 Rückgabeobjekt 898  
 RuleCollection 673  
 RunNow 375  
 Runspace 235  
 RuntimeException 165

## S

sa 612  
 SAM 746  
 SAMAccountName 753, 756, 761, 779  
 Sapien 236, 238, 500 f.  
 Save-Help 61  
 Save-Module 469  
 Save-VM 809  
 Schablone 953  
 Schalter 36, 930  
 Schattenkopie 556  
 ScheduledJob 375  
 Scheduled Task 369  
 Schema 750, 777  
 – Active Directory 755  
 – WMI 305  
 Schemaabfrage 314  
 SchemaNameCollection 742  
 SchemaNamingContext 775  
 Schleife 150  
 Schlüssel 195  
 Schlüsselattribut  
 – WMI 300  
 Schnittstelle 585, 954, 958  
 – .NET 967  
 Schtasks.exe 370  
 Script 444 f.  
 Script Analyzer 216  
 Scripting.FileSystemObject 292  
 ScriptMethod 343  
 ScriptPaneBackgroundColor 216  
 SDDL 545, 735  
 sealed 956  
 SearchScope 783  
 Secure String 559 f.  
 Security Descriptor 719  
 Security Descriptor Definition Language 263  
 Security Identifier 719, 724, 726, 728  
 Security Service Provider 311  
 Select  
 – PowerShell 92  
 SELECT 313, 411  
 – WQL 313 f., 316  
 SelectNodes() 573 f.  
 Select-Object 16, 65 f., 73, 84, 88, 92, 94, 100 f., 178, 332, 574, 913  
 SelectSingleNode() 573 f.  
 Select-String 50, 71, 564  
 Select-Xml 573 f.  
 Semaphore 722  
 Semikolon 101  
 Send-MailMessage 703 f.  
 Send-SmtpMail 703 f.  
 sequence 387  
 Serialisierung 79, 254  
 Seriennummer 639  
 Server 781  
 ServerCertificateValidation-Callback 708  
 Server Management Objects *siehe* SMO  
 Server Manager 794  
 ServerRemoteHost 228  
 ServerURL 451  
 Service 431  
 ServiceController 254  
 Serviceorientierung 960, 962  
 sessionState 226  
 Set-Acl 719, 731, 735  
 Set-ADAccountPassword 781



- Set-ADGroup 793
- Set-ADObject 782f.
- Set-ADOrganizationalUnit 785
- Set-ADUser 786, 788
- Set-Alias 46
- Set-AppLockerPolicy 672
- Set-AuthenticodeSignature 111
- Set-BPAResult 853
- Set-CimInstance 321, 336
- Set-Clipboard 859
- Set-Content 519, 563, 579, 705
- Set-DataRow 600
- Set-DataTable 495, 600
- Set-Date 141
- Set-DistributionGroup 846
- Set-DnsClientServerAddress 688, 690
- Set-DnsClientServerAddress-Funktion 690
- Set-Executionpolicy 109
- Set-ExecutionPolicy 19f., 103
- Set-FileTime 535
- Set-FirewallProfile 698
- Set-GPIInheritance 805
- Set-GPLink 801
- Set-GPPermissions 806
- Set-GPPrefRegistryValue 805
- Set-GPRegistryValue 805
- SetInfo() 740, 749f.
- Set-Item 267, 519
- Set-ItemProperty 534, 636, 703
- Set-JobTrigger 374
- Set-Location 41, 195, 519, 633
- Set-Mailbox 846
- Set-Methode 83
- Set-NetFirewallPortFilter-Funktion 697
- Set-NetFirewallProfile-Funktion 697
- Set-NetFirewallRule-Funktion 697
- Set-NetIPInterface 688
- Set-ODBCDriver 627
- Set-ODBCDsn 627
- Set-PrintConfiguration 649
- Set-PSBreakpoint 356f.
- Set-PSDebug 121, 353f.
- Set-PSSessionConfiguration 261
- Set-ScheduledJob 375
- Set-Service 681, 684
- Set-StrictMode 121
- Set-TargetResource 458
- Setter 83, 966
- Set-TraceSource 356
- Set-Variable 118, 124, 864f.
- Set-VHD 820
- Set-VM 809, 811
- Set-VMMemory 831
- Set-VolumeLabel 524
- Set-WmiInstance 321, 335
- Set-WSManQuickConfig 250
- Shell 3, 65
- Shell.Application 529, 543
- ShouldProcess() 921f.
- Show-Command 57f., 212
- Show-EventLog 246, 712
- Show-HyperVMenu 829
- Show-NetFirewallRule-Funktion 697
- Show-Service 246
- Show-VMMenu 829
- Shutdown 512
- Sicherheit
  - COM 310
  - Dateisystem 297, 307
  - PowerShell 109
  - WMI 310
- Sicherheitsabfrage 921
- Sicherheitsbeschreibung 719
- Sicherheitseinstellung 719
- Sicherheitsmodell 3
- Sicherheitsrichtlinie 110
- SID 719
- Side-by-Side Executing 962
- Signieren 111
- SilentlyContinue 163, 529, 787
- Simple Network Management 296f., 307
- Simple Object Access Protocol 245
- Sitzung 260ff.
- Skip 73
- SkipNetworkProfileCheck 251, 702
- Skript 103, 105
  - PowerShell 103
- Skriptausführungsrechte 18
- Skriptausführungsrichtlinie 19
- Skriptblock 121, 253, 864
- Skriptdatei 103
- Skripteigenschaft 82, 86
- Skriptmodul 939
- SMO 604, 610f., 621, 623, 625
- Smoking Man 738
- SMTP 703f.
- SmtplibClient 703
- Snap-In 465, 476, 887, 889ff., 913, 924
- Snapshot
  - Hyper-V 824
- SNA Server 307
- Snippet 212
- SOAP 306, 706
- Software 297, 500
  - installieren 454, 457, 654
  - inventarisieren 651
  - verwalten 651
- Softwareentwickler 275
- Softwareentwicklungsplattform 961
- Softwarekomponente 286
- Softwarepaket 658
- Softwarequelle 658
- Software Restriction Policy 671
- Sortieren 93
- Sort-Object 65ff., 88, 93f., 100f., 157, 890
- Speech 171
- SpeechSynthesizer 189
- Speicher 72
- Speicherbereinigung 962
- Speicherverbrauch 586
- Speicherverwaltung 962
- Spitzname 953
- Split 132
- Spoolerdienst 649
- Spooling 649
- Sprachausgabe 171, 189
- Sprache 426
- Sprachkürzel 426
- SQL 313, 630
- SQLASCOMMANDETS 603
- Sqlcmd.exe 609
- SqlCommand 590, 609
- SqlConnection 279, 588, 592, 594
- SqlDataSourceEnumerator 584
- SQLPS 197, 602, 604
- SQLPSX 602, 604, 613f., 621
- SQL Server 603
  - Agent *siehe* SMO
  - Laufwerk 605
- SqlServerCe 582
- SqlServerCmdletSnapin100 603
- SQL Server Management Studio 605, 621
- SSL 225, 708
- STA 860
- StackPanel 865
- Stammzertifizierungsstelle 112
- Standarddrucker 187
- Standardkonsole 203
- Start-DscConfiguration 436
- Start-Job 363ff.
- Startmenü 297
- Start-Process 100, 369, 677f.
- Start-PSSession 367
- Start-Service 255, 681, 683
- Start-Sleep 113
- Start-Transaction 359ff.
- Start-Transcript 385



- Startup 512
- Start-VM 809, 818, 831
- Start-WBBackup 558
- Start-WBFileRecovery 558
- Start-WBHyperVRecovery 558
- Start-WBSystemStateRecovery 558
- Start-WBVolumeRecovery 558
- Start-Webitem 843
- Start-Website 843
- static 168
- Status 649
- Stop 163
- Stop-Computer 246, 642
- Stop-Job 363, 366
- Stop-Process 76, 100, 385, 677, 680, 911
- StopProcessing() 889
- Stop-Service 34, 681, 683
- Stop-VM 809
- Stop-WBJob 558
- Stop-Webitem 843
- Stop-Website 843
- Stored Procedure 595
- Streaming 68
- StreetAddress 779
- Subnetzmaske 688
- SubTree 783
- SUBTREE 761
- Suche
  - Active Directory 761
  - Assembly 535
  - LDAP 742
  - Verzeichniseintrag 751
  - XML 573
- SupportsShouldProcess 921
- Surname 779
- Suspend 39
- Suspend-PrintJob 649
- Suspend-Service 681, 683
- Suspend-VM 809
- Switch 36, 146, 152, 807
- SwitchParameter 930
- Sybase 583
- Symbolic Link 538, 540
- SymLink 540f.
- Synopsis 884
- System 963f.
- System32 101
- System ACL 724
- System.ApplicationException 165
- Systemattribut
  - WMI 300
- System.Boolean 199
- System Center Virtual Machine Manager 498
- System.Collections.Hashtable 143
- System.Console 284
- System.Data 277
- System.Data.Odbc 582, 630
- System.Data.OleDb 582, 589
- System.Data.OleDb 582
- System.Data.OracleClient 582, 589
- System.Data.SqlClient 582, 589, 609
- System.Data.SqlClient.SqlConnection 279
- System.Data.SqlServerCe 582
- System.DateTime 118, 140, 279f., 283
- System.Diagnostics.EventLog 711
- System.Diagnostics.Process 67, 75, 79, 175, 677, 909
- Systemdienst 69, 297, 681
  - auflisten 314
  - überwachen 316
- System.DirectoryServices 277, 737ff., 741, 743f., 747, 752, 755, 761, 796
- System.DirectoryServices.ActiveDirectory 796
- System.Directoryservices.DirectoryEntry 278, 281
- System.dll 964
- System-DSN 629
- Systemende 512
- System.Enum 289
- System.Environment 252, 639, 893f., 898
- SystemEvent 412
- System.Globalization.CultureInfo 504
- System.Int32 119, 126
- System.IO.Compression 543
- System.IO.Directory 723
- System.IO.DirectoryInfo 343, 525
- System.IO.DriveInfo 282, 285, 288, 520, 522
- System.IO.DriveType 288
- System.IO.File 723
- System.IO.FileInfo 343, 525, 529
- Systemklassen
  - WMI 299
- Systemmanagement 295
- System.Management 277, 900
- System.Management.Automation 62, 889, 891, 900, 936
- System.Management.Automation.Cmdlet 889
- System.Management.Automation.PathInfo 199f.
- System.Management.Automation.PSCustomObject 565
- System.Management.Automation.PSDriveInfo 520
- System.Management.ManagementObject 141
- System Management Server 307
- System.Media.SoundPlayer 282
- System.Net.WebClient 280, 705, 707
- System.Object 79, 345, 836, 909, 913
- SystemParametersInfo 349
- System.Random 127, 278f.
- System.Reflection 286
- System.Security 724
- System.Security.AccessControl 722
- System.ServiceProcess.ServiceController 79, 681f., 909
- Systemstart 512
- System.String 127, 130, 893
- System.TimeSpan 140
- System.Type 78, 123, 185
- Systemwiederherstellung 643
- System.Windows 860
- System.Windows.FontStyle 863
- System.Windows.Forms 286, 530, 855
- System.Xml.Node 575
- Sysvol 512

## T

- Tab Completion 205
- Tabellenform 184
- TabPanel 865
- Tabulator 129
- Tabulatorvervollständigung 205
- TAR 542
- TaskScheduler 494
- TCP/IP 688
- Tee-Object 97f.
- Telnet 252
- Terminal Services 298
- Terminating Error 160, 917
- Terrabyte 126
- Test-32Bit 646
- Test-64Bit 646
- Test-AppLockerPolicy 672, 674
- Test-Assembly 535
- Test-Connection 24, 695f.
- Test-DbConnection 599
- Test-DscConfiguration 437

Test-ModuleManifest 466  
 Test-Path 199  
 Test-PswaAuthorizationRule 226  
 Test-ServiceHealth 845  
 Test-SqlScript 613  
 Test-TargetResource 458  
 Test-UserGroupMembership 760  
 Test-VHD 810, 820  
 Test-Xml 572  
 Textanzeige 868  
 Textdatei 101, 563  
 Thread 390  
 Thread-Modell 860  
 throw 146, 165  
 ThrowTerminatingError() 917  
 TIFF 530  
 TimeSpan 355  
 TLS 708  
 Ton 283  
 ToString() 79, 909, 957  
 TotalProcessorTime 185  
 TPM 559  
 Trace-Command 514  
 Transaktion 359  
 Transformation 868  
 Translate() 727  
 trap 146, 160, 165  
 Trap 160, 166  
 Treiber 500  
 – ODBC 628  
 Trigger 373  
 Troubleshooting Pack 480, 849  
 true 646  
 Trusted Host 267  
 Trusted Platform Module *siehe*  
 TPM  
 Trustee 720  
 Try...Catch 788  
 Try-Catch-Finally 160, 165  
 T-SQL 609  
 Typ 964  
 – Namensgebung 964  
 Typbezeichner 118  
 types.ps1xml 47f., 86, 88  
 Typisierung 118

## U

UAC *siehe* Benutzerkonten-  
 steuerung  
 Überladung 159  
 Umgebungsvariable 297  
 Umlaut 564  
 Umleitung 188  
 Undefined 110  
 Undo-Transaction 359, 361f.  
 UniformGrid 865

Uninstall-Package 659  
 Universal Coordinated Time 301,  
 336  
 Unix 3, 65f., 115, 519, 538  
 Unlock-BitLocker 562  
 Unregister-PSSessionConfigu-  
 ration 261, 264  
 Unrestricted 110  
 Unterbrechungsfreie Strom-  
 versorgung *siehe* USV  
 Unternehmensraum 963  
 Unterordner 79, 525  
 Unterroutine 154  
 Unterschlüssel 195  
 until 146  
 Update 297, 500  
 Update-Help 60f.  
 UsePropertyCache 749  
 User 431, 743, 752, 765  
 user32.dll 349  
 User Account Control *siehe*  
 Benutzerkontensteuerung  
 UseTestCertificate 225  
 UseTransaction 360  
 using 388  
 USV 647  
 UTF8 564

## V

ValidateCount 882  
 Validate-CustomerID 882  
 ValidateLength 124, 882, 930  
 ValidateNotNull 882, 930  
 ValidatePattern 124, 882, 930  
 ValidateRange 124, 882  
 ValidateScript 124, 882  
 ValidateSet 124  
 ValueFromPipeline 882, 906,  
 908, 913  
 ValueFromPipelineByProperty-  
 Name 882, 908  
 ValuesCollection 741  
 Variable 78, 98, 117, 123, 180, 195  
 – Auflösung 128  
 – Workflow 389  
 Variablenuflösung 128, 180  
 Variablenkennzeichner 98, 117  
 VB 347, 349  
 Verbindungszeichenfolge 279,  
 588, 595  
 Verbose 40, 437, 916  
 VerbosePreference 916  
 VerbsCommon 929  
 VerbsCommunications 929  
 VerbsData 929  
 VerbsDiagnostic 929

VerbsLifecycle 929  
 VerbsSecurity 929  
 Vererbung 305, 956, 967  
 Vererbungsdiagramm 956  
 Vererbungshierarchie 318, 777,  
 956  
 – WMI 305  
 Vergleich 99  
 Vergleichsoperator 88  
 Verifikation 962  
 Verknüpfung 537  
 Verzeichnisattribut 748  
 Verzeichnisdienst 87, 307, 495,  
 750, 762  
 Verzeichnisdienstklasse 750  
 Verzeichnisobjekt 747, 752  
 Verzweigung 97  
 VHD 819f., 830  
 VHDX 816, 820, 830  
 Video 868  
 View 176  
 VirtualHardDisk 820  
 Virtualisierung 807  
 VirtualizingStackPanel 865  
 Virtuelle Maschine 499, 807, 814  
 Virtuelles System 807  
 Virus 110  
 Visual Basic 347, 959  
 Visual Basic 6.0 962  
 Visual Basic .NET 887f.  
 Visual Studio 239f., 393, 887f.,  
 913  
 Visual Web Developer Express  
 888  
 VMBus 270  
 VMGUID 270  
 VMName 270  
 VM *siehe* Virtuelle Maschine  
 void 187  
 VolumeLabel 281  
 Volume Shadow Copy Service  
 556  
 VSS *siehe* Volume Shadow Copy  
 Service

## W

Wait 437  
 WaitForAll 458  
 WaitForAny 458  
 WaitForSome 458  
 Wait-Job 363, 366  
 Wait-Process 680  
 Walter Skinner 738  
 WarningAction 40, 163  
 WarningVariable 40  
 Warnung 40

- WAS *siehe* Windows Activation Service
- WBEM 295
- WDAC *siehe* Windows Data Access Components
- Web Administration 480
- WebAdministration 833, 835
- Webanwendung 961
- Web Based Enterprise Management 295 f.
- Webdienst 706
- Weblog 705, 973
- Webserver 197, 838
- Webservice 708
- Web Service Description Language 707
- Webservices 706
- Website 838, 844
- Well-Known GUID 746
- Well-Known Object 746
- Well-Known Security Principal 728
- WellKnownSidType 729
- Werkzeug 203
- Wertemenge 141
- whatif 650
- Whatif 39
- WhatIf 527, 921
- WHERE 313
- Where-Object 42, 65 ff., 76 f., 88, 90, 96, 100 f., 187, 681, 890, 901, 913
- while 146
- Whistler 303
- Width 184
- Wiederherstellungspunkt 643
- Win32 298
- Win32\_Account 739
- Win32\_ACE 547 f.
- Win32-API 349
- Win32\_Battery 647
- Win32\_Bios 640
- Win32\_BootConfiguration 640
- Win32\_CDRomDrive 645
- Win32\_CDRomDrive 332
- Win32\_CodecFile 653
- Win32\_ComponentCategory 314
- Win32\_ComputerShutdownEvent 311, 412
- Win32\_Computersystem 639
- Win32\_currenttime 141
- Win32\_Currenttime 141
- Win32\_Desktop 739
- Win32\_Diskdrive 645
- Win32\_Group 739
- Win32\_Keyboard 646
- Win32\_LocalTime 141
- Win32\_LocalDisk 523
- Win32\_LogicalDisk 305, 314, 337, 520, 522, 900
- Win32\_MappedLogicalDisk 524
- Win32\_MemoryDevice 645
- Win32\_NetworkAdapter 646
- Win32\_NetworkAdapterConfiguration 314, 685, 691 f.
- Win32\_NTLogEvent 314, 316, 413
- Win32\_OperatingSystem 639
- Win32\_OSRecoveryConfiguration 640
- Win32\_PerfRawData 715
- Win32\_PerfRawData\_PerfOS\_Processor 715
- Win32\_PerfRawData\_PerfProc\_Process 715
- Win32\_PingStatus 695
- Win32\_PointingDevice 646
- Win32\_PowerManagementEvent 412
- Win32\_Printer 647, 649, 685
- Win32\_Printjob 647 f.
- Win32\_Process 413
- Win32\_Processor 645
- Win32\_ProcessStartTrace 412
- Win32\_Product 651, 654 f.
- Win32\_Quickfixengineering 653
- Win32\_SecurityDescriptor 548
- Win32\_Service 314, 316, 413
- Win32\_Share 545 f.
- Win32\_SoundDevice 645
- Win32\_SystemConfigurationChangeEvent 311, 412
- Win32\_Tapedrive 645
- Win32\_TCPIPPrinterPort 647 f., 685
- Win32\_Trustee 547
- Win32\_USBController 646
- Win32\_UserAccount 305, 739
- Win32\_Videocontroller 332
- Win32\_VideoController 322, 645
- Win32\_Volume 524
- Win32\_WindowsProductActivation 640
- Windows 7 208, 479
- Windows 8 327, 833
- Windows 8.0 481
- Windows 8.1 483
- Windows 9x 306
- Windows 10 5, 9, 22, 270, 485
- Windows 2000 303
- Windows Activation Service 682
- Windows as a Service 5
- Windows-Authentifizierung 612
- Windows Communication Foundation 298
- Windows Data Access Components 627
- Windows Driver Model 307
- Windows Explorer 636
- Windows Firewall 500, 696, 701
  - Per PowerShell konfigurieren 696
- Windows-Firewall
  - Im Netzwerk abfragen 702
- Windows Forms 855, 857
- Windows Installer 307, 671
- Windows Management Framework 10, 296, 649
- Windows Management Instrumentation 29
- Windows ME 306
- Windows Nano Server 9
- Windows PowerShell 3
- Windows PowerShell Community Extensions 490
- Windows Pre Installation Environment *siehe* WinPE
- Windows Presentation Foundation *siehe* WPF
- Windows Remote Management 245, 247, 249 f., 307, 364
- Windows Script Host 30, 109
- Windows Server 2003 4, 303, 306, 761
- Windows Server 2008 208
- Windows Server 2008 R2 208
- Windows Server 2012 327, 481, 794, 833
- Windows Server 2012 R2 229, 483
- Windows Server 2016 9, 270, 485
- Windows Server Core 210
- Windows Troubleshooting Platform 849
- Windows Vista 208, 959
- Windows XP 29, 303, 313
- WinMgmt.exe 306 f.
- WinPE 559
- WinRM 307, 642
- WITHIN 313, 412
- WKGUID 746 f.
- WMI 3, 29, 141, 245, 295, 298, 305, 324, 900, 904, 917
  - Class Explorer 318
  - Command Shell 29
  - Data Query 314
  - Ereignis 311
  - Event Query 313, 315
  - Klasse 318
  - Namespace 303
  - Object Browser 317 f.
  - Query Language 313, 326

- Repository 306, 311, 328
- Schema 305, 314
- Schema-Query 314
- Steuerung 306
- WMIClass 324, 326
- WMICLASS 295
- WMI Object Browser 317
- WMI Query Language *siehe* WQL
- WMISEARCHER 295, 324, 326
- Word 94
- Workflow 379, 385 ff., 393
  - Designer 394
  - Einschränkungen 384
  - Persistenz 392
  - Verschachtelt 389
- WorkflowInfo 394
- WorkingSet 87
- WorkingSet64 67
- World Wide Wings 927
- Wörterbuch 94
- WPF 210, 383, 855, 860
- WPF PowerShell Kit 494, 860
- WPK 494, 860
- WQL 313, 327, 411, 900
- WrapPanel 865
- Write-BZip2 542
- Write-Clipboard 859
- WriteDebug() 917
- Write-Error 180
- WriteError() 917, 920
- Write-EventLog 246, 711, 713
- Write-GZip 22, 542
- Write-Host 180, 228, 386
- WriteObject 894, 902

- WriteObject() 890, 900
- Write-Tar 542
- WriteVerbose() 917, 920
- Write-Warn 180
- WriteWarning() 917, 920
- Write-Zip 542
- WScript.Shell 537
- WSDL 706
- WSH 744
- WSMan 197, 269
- WS-Management 245 f., 249,  
268, 307, 319, 321, 323
- Wurzelnamensraum 963
- www.IT-Visions.de 479, 495, 520,  
XXIV

## X

- x64 807
- x86 807
- XAML 383, 393, 868
- XCopy-Deployment 962, 965
- xDscDiagnostics 452
- xDscWebService 447
- XFilesServer 738
- Xml 571
- XML 60, 191, 461, 553, 570, 575,  
577, 707, 923
- XML Application Markup  
Language *siehe* XAML
- XmlAttribute 574
- XmlDocument 577
- XmlElement 574
- XML-Schema 572

- XML-Webservice 962
- XPathDocumentNavigator 574
- XslCompiledTransform 577

## Y

- Year 74

## Z

- Zahl 125
- Zahlenliteral 126
- Zahlfallszahl 127
- Zeichenkette 127 f., 132, 180, 910,  
929
  - Operation 131
  - trennen 132
  - verbinden 132
- Zeichensatz 564
- Zeitmessung 355
- Zeitplandienst 297
- Zertifikat 111, 225
- Zertifikatsspeicher 3, 195
- Zertifikatsverwaltung 112
- ZIP 541, 542
- ZipFile 543
- Zufallszahl 126
- Zugriffsrechtliste 719, 724
- Zugriff verweigert 336
- Zuweisungsoperator 145
- Zwischenablage 859
- Zwischencode 960
- Zwischenschritt 96
- Zwischenspeicher 586