PowerShell Basics

Wiederholungsfragen

DIST IT SERVICES & TRAINING

1

Cmdlets

- **Geben sie alle Cmdlets aus, deren Zeitwort remove ist.**
  - `Get-Command -Verb remove`
- **Geben Sie alle Aliase aus.**
  - `Get-Alias`
- **Erstellen Sie einen Alias ‚sos' für das cmdlet get-help.**
  - `New-Alias -Name sos -Value Get-Help`
  - `New-Alias sos Get-Help`
  - `Set-Alias -Name sos -Value Get-Help`

DIST IT SERVICES & TRAINING

2

Cmdlets

- **Geben Sie die Liste aller Prozesse aus.**
  - `Get-Process`
- **Lassen sie sich alle Prozesse anzeigen, die mehr als 1000 Handles eingerichtet haben.**
  - `Get-Process | Where-Object { $_.Handles -gt 1000 }`
- **Geben sie nur die Prozesse aus, deren Namen mit "d" beginnen. (Hint: Operator –like, Wildcard Character *)**
  - `Get-Process | Where-Object { $_.ProcessName -like "d*" }`
  - `Get-Process d*`
- **Geben Sie die Liste der Prozesse aus, deren Namen mit "i" oder "p" beginnen.**
  - ```
    Get-Process |
        Where-Object { $_.ProcessName -like "i*" -or
                       $_.ProcessName -like "p*" }
    ```
  - `Get-Process | Where-Object {$_.ProcessName -like "[ip]*" }`
  - `Get-Process "[ip]*"`

Cmdlets

- **Lassen Sie sich von den Prozessen nur die ID und den Namen anzeigen.**
  - `Get-Process | Select-Object -Property ID,ProcessName`
  - `Get-Process | Format-Table -Property ID,ProcessName`
- **Welche drei Prozesse benötigen am meisten Speicher?**
  - ```
    Get-Process | Sort-Object -Property WorkingSet -Desc |
                  Select-Object -First 3
    ```

PowerShell 7:

```
Get-Process |
  Sort-Object -Property WorkingSet -Desc -Top 3
```

- **Dateien welchen Typs sind im Windows Verzeichnis?**
  - `Get-Childitem C:\Windows -File | Select-Object Extension -Unique`
- **Listen Sie die notwendigen Files eines Druckertreibers auf.**
  - ```
    Get-PrinterDriver -Name 'Microsoft Print To PDF' |
        Select-Object -ExpandProperty DependentFiles
    ```
- **Wie viel freien Speicher gibt es auf jedem Laufwerk (Volume)? Die Liste soll nur den Laufwerksbuchstaben und den freien Speicher in GB zeigen.**
  - ```
    $FreeSpaceHT = @{ Label=„Size(GB)“;
                      Expression={ $_.SizeRemaining / 1GB } }
    ```
  - `Get-Volume | Select-Object DriveLetter,$FreeSpaceHT`

Cmdlets – File Content

- **Lassen Sie sich den Inhalt der Datei C:\Windows\setupact.log anzeigen.**
  - `Get-Content -Path C:\Windows\setupact.log`
- **Lassen Sie sich nur die ersten 5 bzw. letzten 3 Zeilen der Datei C:\Windows\setupact.log anzeigen.**
  - `Get-Content -Path C:\Windows\setupact.log -TotalCount 5`
  - `Get-Content -Path C:\Windows\setupact.log -Tail 3`
- **Speichern Sie die Ausgabe von Get-ChildItem -Path C:\Windows\ -File in die Datei C:\Temp\Info.txt.**
  - `Get-ChildItem -Path C:\Windows\ -File | Out-File -FilePath C:\Temp\Info.txt`
  - `Get-ChildItem -Path C:\Windows\ -File > C:\Temp\Info.txt`
- **Fügen Sie der Datei C:\Temp\Info.txt eine weitere Zeile 'FINISHED' hinzu.**
  - `Add-Content -Value 'FINISHED' -Path C:\Temp\Info.txt`
  - `'FINISHED' >> C:\Temp\Info.txt`
  - `'FINISHED' | Out-File -FilePath C:\Temp\Info.txt -Append`
- **Löschen Sie den Inhalt der Datei C:\Temp\Info.txt.**
  - `Clear-Content –Path C:\Temp\Info.txt`

DIST IT SERVICES & TRAINING

5

Cmdlets – Explore the PowerShell

- **Wie werden die konfigurierten IP-Adressen angezeigt (ipconfig)?**
  - `Get-NetIPConfiguration`
  - `Get-NetIPAddress`
- **Konfigurieren Sie eine die IP-Adresse 1.2.3.4/16 für die Netzwerkkarte (Idx 7). Default Gateway sei 1.2.0.1**
  - `New-NetIPAddress -InterfaceIndex 7 -IPAddress 1.2.3.4 -PrefixLength 16 -DefaultGateway 1.2.0.1`
- **Wie ändern Sie den DG einer Netzwerkkarte (Idx 7)?**
  - `Remove-NetRoute -InterfaceIndex 7 -DestinationPrefix '0.0.0.0/0'`
  - `New-NetRoute -InterfaceIndex 7 -DestinationPrefix '0.0.0.0/0' -NextHop 10.10.10.250`
  - `Get-NetRoute`
- **Wie konfigurieren Sie die DNS Server Adressen einer Netzwerkkarte (Idx 7)?**
  - `Set-DnsClientServerAddress -InterfaceIndex 7 -ServerAddresses 1.1.1.1,8.8.8.8`
- **Wie löschen Sie den DNS-Client-Cache (ipconfig /flushdns)?**
  - `Clear-DNSClientCache`

DIST IT SERVICES & TRAINING

6

Cmdlets – Explore the PowerShell

- **Mit welchem cmdlet lassen sich „Networkneighbors“ anzeigen (arp –a)?**
  - `Get-NetNeighbor`
- **Wie können Sie die MAC-Adresse einer Netzwerkkarte ändern (devmgmt.msc)?**
  - `Set-NetAdapter -Name Ethernet -MACAddress aa-bb-cc-dd-ee-ff`
- **Mit welchem cmdlet lässt sich eine VPN Verbindung löschen (control.exe)?**
  - `Remove-VpnConnection -Name xyz`

DIST IT SERVICES & TRAINING

7

Cmdlets – Explore the PowerShell

- **Wie können Sie die verfügbaren Rollen und Funktionen (Features) eines Windows Servers sehen?**
  - `Get-WindowsFeature`
  - `Get-WindowsFeature | Where-Object { $_.InstallState -eq „Available“ }`
  - `Get-WindowsFeature | Where-Object { $_.Installed -eq $true }`
- **Wie kann die Rolle DHCP am Windows Server installiert werden?**
  - `Install-WindowsFeature -Name DHCP -IncludeManagementTools`
- **Mit welchem cmdlet lassen sich aktuelle Druckaufträge anhalten?**
  - `Suspend-PrintJob`
  - `Get-PrintJob -PrinterName XYZ | Suspend-PrintJob`
- **Mit welchem cmdlet können Sie eine neue Freigabe erstellen?**
  - `New-SMBShare`

DIST IT SERVICES & TRAINING

8

Cmdlets – Strings

- **Erzeugen Sie die folgenden Variablen:**
  - `$hostname = 'ServerX'`
  - `$Amount = 5`
  - `$OS = 'WinSrv2022'`
- **Unter Verwendung der o. a. Variablen lassen Sie sich folgenden Text ausgeben:**

The host ServerX provides 5 enduser services on the OS WinSrv2022.

  - `'The host ' + $hostname + ' provides ' + $Amount + ' enduser services on the OS ' + $OS + '.'`
  - `"The host $hostname provides $Amount enduser services on the OS $OS."`
  - `"The host {0} provides {1} enduser services on the OS {2}." -f $hostname, $Amount, $OS`

DIST IT SERVICES & TRAINING

9

Cmdlets

- **Ermitteln Sie die Eigenschaften, Methoden etc. von Systemdiensten.**
  - `Get-Service | Get-Member`
- **Ermitteln Sie *nur* die Eigenschaften von Systemdiensten.**
  - `Get-Service | Get-Member -MemberType Property`
- **Geben Sie eine Liste der laufenden Dienste aus.**
  - `Get-Service | Where-Object { $_.status -eq "running" }`
- **Ändern Sie den StartupType vom Dienst "Browser" auf "Manual".**
  - `Get-Command *service`
  - `Get-Help Set-Service -detailed`
  - `Set-Service Browser -startuptype manual`

DIST IT SERVICES & TRAINING

10

Cmdlets

- **Wie viele cmdlets haben das Verb "get" oder "set". Geben Sie nur die Zahl aus.**
  - `$result = Get-Command -Verb get,set | Measure-Object`
  - `$result = Get-Command "[sg]et-*" | Measure-Object`
    `$result.count`
  - `(Get-Command "[sg]et-*" | Measure-Object).Count`
  - `(Get-Command "[sg]et-*").Count`
- **Wie groß ist die größte Datei im Verzeichnis C:\Windows und seinen Unterverzeichnissen und wie viel Speicher benötigen alle Dateien in Summe?**
  - `$files = Get-Childitem 'C:\Windows' -Recurse`
    `$files | Measure-Object Length -Max -Sum`

DIST IT SERVICES & TRAINING

11

Cmdlets

- **Lassen Sie sich anzeigen, welche cmdlets Aliase haben und wie viele es sind.**
  - `Get-Alias | Group-Object Definition`
- **Welches cmdlet hat die meisten Aliase und wie lauten sie?**
  - `Get-Alias | Group-Object Definiton | Sort-Object count -descending | Select-Object Group -First 1 -ExpandProperty Group`
- **Von welchem Dateityp gibt es die meisten Dateien in C:\Windows und seinen Unterverzeichnissen?**
  - `$files = Get-Childitem 'C:\Windows' -Recurse`
  - `$files | Group-Object Extension | Sort-Object Count -Descending | Select-Object Name -First 1`

DIST IT SERVICES & TRAINING

12

Variablen & Arrays

- **Speichern Sie den Wert 12,34 in einer Variable mit dem Namen "Eingabe".**
  - `$Eingabe = 12.34`
- **Speichern Sie den Wert 34,55 in einer Variable vom Typ "Double" mit dem Namen "Zielwert".**
  - `[double]$Zielwert = 34.55`
- **Lassen Sie sich alle Variablen anzeigen und löschen Sie im Anschluss die Variable $Eingabe.**
  - `Get-Variable`
  - `Remove-Variable Eingabe`

DIST IT SERVICES & TRAINING

13

Variablen & Arrays

- **Speichern Sie die Werte "Graz", "Linz" und "Bregenz" im Array Cities. Stellen Sie sicher, dass im Array nur Strings gespeichert werden können.**
  - `[String[]]$Cities = "Graz", "Linz", "Bregenz"`
- **Fügen Sie dem Array $Cities den Wert "Innsbruck" hinzu.**
  - `$Cities += "Innsbruck"`
- **Stelle Sie fest, ob sich im Array der Wert "Wien" befindet.**
  - `"Wien" -in $Cities`
  - `$Cities -contains "Wien"`

DIST IT SERVICES & TRAINING

14

Variablen & Arrays

- **Wie viele Werte sind aktuell im Array $Cities gespeichert?**
  - `$Cities.count`
- **Erzeugen Sie ein neues Array mit dem Namen $LHS. Es soll die gleichen Werte wie $Cities beinhalten. Löschen Sie anschließend das Array $Cities.**
  - `$LHS = $Cities`
  - `Remove-Variable Cities`

DIST IT SERVICES & TRAINING

15

Hashtable

- **Erzeugen Sie folgende Hashtable:**
  - `$Mitarbeiter = @{ Name="Franz";Abteilung="Personal";Dienstjahre=15 }`

```
PS C:\> $Mitarbeiter

Name                           Value
----                           -----
Dienstjahre                    15
Name                           Franz
Abteilung                      Personal


PS C:\>
```

- **Lassen Sie sich nur den Inhalt von Abteilung anzeigen.**
  - `$Mitarbeiter["Abteilung"]`
  - `$Mitarbeiter.Abteilung`

DIST IT SERVICES & TRAINING

18

Hashtable

- **Fügen Sie den Wert $False unter "Prokurist" hinzu.**
  - `$Mitarbeiter.Add("Prokurist",$False)`
  - `$Mitarbeiter = $Mitarbeiter + @{ Prokurist = $False }`
  - `$Mitarbeiter += @{ Prokurist = $False }`
  - `$Mitarbeiter["Prokurist"] = $False`
  - `$Mitarbeiter.Prokurist = $False`
- **Entfernen Sie den Wert "Abteilung".**
  - `Get-Help about_Hash_Table`
  - `$Mitarbeiter.Remove("Abteilung")`
- **Sortieren Sie die Hashtable nach den Wertnamen absteigend.**
  - `$Mitarbeiter.GetEnumerator() | Sort-Object Key -descending`
  - *Hint:* `$Mitarbieter.GetEnumerator() | Sort-Object Value`

19

Cmdlets – Format-*

- **Erstellen Sie eine Tabelle aller Netzwerkadapter mit den Eigenschaften "Name", "Status","Hidden","FullDuplex","MACAddress" und "InterfaceIndex"**
  - `Get-NetAdapter | Format-Table -Property Name, State, Hidden, ↵ FullDuplex, MACAddress, InterfaceIndex`
- **Lassen Sie sich die verwendeten Dateierweiterungen des System32-Verzeichnisses in 5 Spalten anzeigen.**
  - `Get-Childitem c:\Windows\System32 -File | Select-Object Extension -Unique | Format-Wide Extension -Column 5`
- **Zeigen Sie alle Eigenschaften des Admin$-Shares an.**
  - `Get-SmbShare -Name Admin$ | Format-List -Property *`
  - `Get-SmbShare -Name Admin$ | Format-List *`
  - `Get-SmbShare -Name Admin$ | fl *`

20

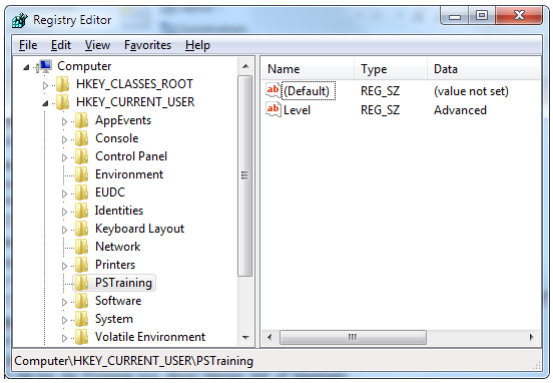
Cmdlets

- **Geben Sie eine Liste aller Dateien aus dem Verzeichnis 'c:\program files' aus, die größer als 1 MB sind.**
  - `Get-Childitem "c:\program files" -File -Recurse |`
    `Where-Object { $_.Length -gt 1MB }`
- **Geben Sie eine Liste aller Dateien aus dem Verzeichnis 'c:\program files' aus, die größer als 1 MB sind.**
  Die Liste soll nur die Dateinamen und die Größe in MB enthalten.
  - `Get-Childitem "c:\program files" -File -Recurse |`
    `Where-Object { $_.Length -gt 1MB } |`
    `Format-Table Name,@{ Name="Length";`
    `Expression={ $_.Length/1MB};`
    `FormatString="n2"`
    `}`
  - *Oder:* `… @{Name='Length';`
    `Expression={("{0:n2}MB" -f ($_.Length/1MB) )}`
    `}`

21

Registry

- **Erzeugen sie folgende Struktur in der Registry mit der PowerShell.**
  - `New-Item -Path HKCU:\ -Name PSTraining -Type Key`
  - `New-Itemproperty -Path HKCU:\PSTraining -Name Level`
    `-Propertytype String -Value "Advanced"`



22

Registry

- **Welcher Wert ist in "Version" unter HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectX gespeichert?**
  - `(Get-Itemproperty -Path HKLM:\Software\Microsoft\DirectX).Version`
  - `(Get-Item HKLM:\Software\Microsoft\DirectX).GetValue("Version")`
  - `Get-ItempropertyValue -Path HKLM:\Software\...\DirectX -Name Version`
- **oder: Welcher Wert ist in "UseSep" unter HKEY_CURRENT_USER\SOFTWARE\Microsoft\Calc gespeichert?**
  - `(Get-Itemproperty -Path HKCU:\Software\Microsoft\Calc).UseSep`
  - `(Get-Item HKCU:\Software\Microsoft\Calc).GetValue("UseSep")`
  - `Get-ItempropertyValue -Path HKCU:\Software\Microsoft\Calc -Name UseSep`

DIST IT SERVICES & TRAINING

23

Daten Im- & Export

- **Speichern Sie von den laufenden Prozessen, jedoch ohne svchosts, die PID, Namen und Handlecount in eine csv-Datei. Verwenden sie als Trennzeichen einen Strichpunkt ';' .**
  - ```
    Get-Process | Where-Object { $_.ProcessName -ne "svchost" } |
          Select-Object ID,ProcessName,Handlecount |
          Export-Csv c:\temp\processes.csv -Delimiter ";"
    ```
- **Starten Sie 3x Notepad und hängen Sie die Processinformationen dem File processes.csv an.**
  - `Notepad; Notepad; Notepad #oder: 1..3 | Foreach-Object { notepad }`
  - ```
    Get-Process notepad |
          Export-Csv c:\temp\processes.csv -Delimiter ";" -Append
    ```
- **Importieren Sie die Daten der csv-Datei in die Variable $proc und lassen Sie sich den 7. Eintrag anzeigen.**
  - `$proc = Import-Csv c:\temp\processes.csv -Delimiter ";"`
  - `$proc[6]`

DIST IT SERVICES & TRAINING

25

Enumeration

- **Kopieren Sie alle *.cpl Dateien aus dem Verzeichnis C:\Windows\System32 in das Verzeichnis C:\Temp und lassen Sie sich anzeigen, welche Dateien kopiert wurden.**
  Hinweis: Verwenden Sie dafür die folgenden cmdlets:
  Get-Childitem - Foreach-Object - Copy-Item - Write-Host
  - `Get-Childitem C:\Windows\System32\*.cpl |`
    `Foreach-Object { Copy-Item $_ -Destination C:\Temp;`
    `Write-Host „File $($_.Name) wurde kopiert.“ }`
- **Erweitern Sie die obige Lösung dahingehend, dass zusätzlich die Anzahl der kopierten Dateien angezeigt wird.**
  - `Get-Childitem C:\Windows\System32\*.cpl |`
    `Foreach-Object –Process { Copy-Item $_ -Destination C:\Temp;`
    `Write-Host „File $($_.Name) wurde kopiert.“;`
    `$i++ }`
    `-Begin { $i = 0 }`
    `–End { „Es wurden $i Dateien kopiert.“ }`

Eventlogs

- **Lassen Sie sich die 15 letzten Events im Application Log anzeigen.**
  - `Get-Eventlog –Logname Application –Newest 15`
- **Lassen Sie sich alle Events im Application Log vom gestrigen Tag anzeigen.**
  - `Get-EventLog -LogName Application`
    `-After (Get-Date).AddDays(-1).Date`
    `-Before (Get-Date).Date`
  - `Get-Eventlog –Logname Application |`
    `Where-Object { $_.TimeGenerated.Date –eq (Get-Date).Adddays(-1).Date }`

Eventlogs

- **Lassen Sie sich 10 Events des Microsoft-Windows-GroupPolicy/Operational Log anzeigen.**
  - `Get-WinEvent -Logname Microsoft-Windows-GroupPolicy/Operational -maxevents 10`
  - `BUG bis zur Version 3.0; Culture auf en-US in Systemsteuerung umstellen.`

DIST IT SERVICES & TRAINING

28

Eventlogs

- **Schreiben Sie ein Event in das selbsterstellte Log PSTraining und verwenden Sie dazu die Quelle Shell.**
  - `New-EventLog -Logname PSTraining -Source Shell`
  - `Write-EventLog -Logname PSTraining -Source Shell -Message "xyz" -EntryType Information -Id 2345`

DIST IT SERVICES & TRAINING

29

WMI

- **Lassen Sie sich anzeigen, wie viel RAM eingebaut ist.**
  - `(Get-CimInstance Win32_ComputerSystem).TotalPhysicalMemory`
- **Seit wann ist der Rechner eingeschaltet?**
  - `(Get-CimInstance Win32_OperatingSystem).LastBootUpTime`
- **Wie viele commands werden automatisch bei einem User Login gestartet?**
  - `Get-CimInstance Win32_StartupCommand | Measure-Object`
- **Lassen Sie sich die Methoden und Properties der Classes Win32_Product anzeigen.**
  - `(Get-CimClass -ClassName Win32_Product).CimClassProperties | Format-Table`
  - `(Get-CimClass -ClassName Win32_Product).CimClassMethods | Format-Table`
- **Welche Produkte von Adobe sind auf Ihrem System installiert?**
  - ```
    Get-CimInstance –ClassName Win32_Product |
                    Where-Object { $_.Vendor –like "*Adobe*" }
    ```

DIST IT SERVICES & TRAINING

30

WMI

- **Wie lässt sich 'Adobe Acrobat (64-bit) von einem System deinstallieren?**
  - ```
    $AR = Get-CimInstance –ClassName Win32_Product |
                    Where-Object { $_.Vendor –eq "Adobe Acrobat (64-bit)" }
    $AR | Invoke-CimMethod -MethodName Uninstall
    ```

DIST IT SERVICES & TRAINING

31

AD DS

Verwenden Sie die VMs im Az-040 Lab – Lab 2!

- **Exportieren sie alle User der OU 'Managers' in ein xml-File (ADUsers.xml). Folgende sechs Attribute sollen gespeichert werden:**
  - *samAccountName, Name, DistinguishedName*
  - *JobTitle, Department, SID*

```
Get-ADUser -SearchBase "ou=Managers,dc=adatum,dc=com" -Filter *
           -Properties Department,Title |
    Select-Object Name, Distinguishedname,
           SAMAccountName, SID, Title, Department |
    Export-Clixml C:\Temp\ADUsers.xml
```

- **Stellen Sie sicher, dass eine etwaiges vorhandenes xml-File nicht überschrieben wird.**
  - … -NoClobber

33

AD DS

Verwenden Sie die VMs im Az-040 Lab – Lab 2!

- **Erzeugen Sie in der OU Sales ein Security-Group (universal) mit dem Namen 'Promotor'.**
  - `New-ADGroup –Path ,OU=Sales,OU=Marketing,DC=Adatum,DC=com' –Name 'Promotor' -SAMAccountname 'Promotor' -GroupCategory Security –GroupScope Universal`
- **Fügen Sie die Benutzer**
  Kerri West
  August Towle
  Julio Bevins und
  Dixie Ferrell
  als Mitglieder der Gruppe 'Promotor' hinzu.
  - `Add-ADGroupMember –Identity Promotor –Members Kerri,August,Julio,Dixie`

34

AD DS

Verwenden Sie die VMs im Az-040 Lab – Lab 2!

- **Legen Sie für Amy Pond in der OU 'Marketing' ein aktives (enabled) Benutzerkonto an. Vergeben Sie das Passwort ‚Schallschrauber!'. Der Anmeldename sei Amy. Für den UserprincialName verwenden Sie das Domänen-Suffix 'adatum.com'.**
  -
```
New-ADUser –Path 'OU=Marketing,DC=Adatum,DC=com' –Name 'Amy Pond'
      -SamAccountName Amy –UserPrincipalName 'amy.pond@adatum.com'

$Password = Convertto-SecureString 'Schallschrauber!' –AsPlainText -Force

Set-ADAccountPassword –Identity Amy –NewPassword $Password

Enable-ADAccount –Identitiy Amy
```
  -
```
$Password = Convertto-SecureString 'Schallschrauber!' –AsPlainText –Force
New-ADUser –Path 'OU=Tardis,DC=Account,DC=com' –Name 'Amy Pond'
             -SamAccountName Pond –UserPrincipalName 'pond@adatum.com'
             -AccountPassword $Password –Enabled $true
```

DIST IT SERVICES & TRAINING

35

Remoting

*Verwenden Sie vom Az-040 Lab – Lab 2 die VM **Lon-CL1**!*

- **Lassen Sie sich die Freigaben vom Server 'lon-dc1' anzeigen.**
  - `Invoke-Command –Computername lon-dc1 –Scriptblock { Get-SMBShare }`
- **Lassen Sie sich alle Prozesse der Server 'lon-dc1' und 'lon-svr1' anzeigen, die mehr als 50MB RAM benötigen.**
  -
```
Invoke-Command –Computername lon-dc1,lon-svr1 `
               –ScriptBlock { Get-Process | Where-Object { $_.Workingset –gt 50MB } }
```
- **Richten Sie eine permanente Session zum Server 'lon-dc1' ein und verwenden Sie diese um den freien Platz auf dem Laufwerk c: zu ermittlen.**
  - `$dc1session = New-PSSession –Computername lon-dc1`
  - `Invoke-Command –Session $dc1session –Scriptblock { Get-Volume –DriveLetter c }`
- **Unter Verwendung der Variable $dc1session, importieren Sie das Modul DHCPServer'.**
  - `Import-Module –Name DHCPServer –PSSession $dc1session`
- **Unter Verwendung der Variable $dc1session, starten Sie eine remote Shell am Server 'lon-dc1'.**
  - `Enter-PSSession -Session $dc1session`

DIST IT SERVICES & TRAINING

39

Jobs

*Verwenden Sie vom Az-040 Lab – Lab 2 die VM **Lon-CL1**!*

- **Generieren Sie einen Background Job mit dem Namen 'BigFiles', der alle Dateien auf C:\ ermittelt, die größer als 10MB sind.**
  - `Start-Job –ScriptBlock { Get-Childitem c:\ -Recurse | Where-Object { $_.Length –ge 10MB } } –Name 'BigFiles'`
- **Lassen Sie sich alle Jobs anzeigen.**
  - `Get-Job`
- **Wie kann die Liste der gefundenen Files in der Variable $BigFiles gespeichert werden?**
  - `$BigFiles = Receive-Job -Name BigFiles -Keep`
- **Generieren Sie einen Remote Backgroud Job mit dem Namen 'BigRemoteFile', der alle Dateien auf C:\ ermittelt, die größer als 10MB sind und sich am Server 'Lon-DC1' befinden.**
  - `Invoke-Command -ScriptBlock { Get-ChildItem c:\ -Recurse | Where-Object { $_.Length -ge 10MB } } -ComputerName Lon-DC1 -AsJob -JobName BigRemoteFile`

DIST IT SERVICES & TRAINING

40