

10 | 19

Sonderdruck für ScriptRunner Software

IT

Administrator

Das Magazin für professionelle System- und Netzwerkadministration

Exchange mit der PowerShell verwalten





Exchange mit der PowerShell verwalten

Abwesenheitsmanagement

von Heiko Brenn

Bei der Verwaltung von Exchange-Servern spielt die PowerShell eine immer wichtigere Rolle. Das liegt zum einem daran, dass eine ganze Reihe von Funktionen erst gar nicht über das Exchange Admin Center zur Verfügung stehen, zum anderen an der Tatsache, dass die Nutzung der Admin-GUI für regelmäßig wiederkehrende Tasks viel Zeit in Anspruch nimmt. In diesem Workshop schauen wir uns an, mit welchen Cmdlets sich das Abwesenheitsmanagement in Exchange umsetzen lässt.

Mit Hilfe der PowerShell können Sie in Exchange 2019 viele Aufgaben einfach, schnell und standardisiert erledigen. Das Thema Abwesenheitsmanagement ist hierfür ein gutes Beispiel, denn nicht nur zur Urlaubszeit gerät das Einrichten beziehungsweise Anpassen der Out-of-Office-Einstellungen für einzelne oder mehrere Mitarbeiter immer wieder zur Herausforderung in den Administrations- und Helpdesk-Teams.

Obwohl wir uns in diesem Workshop mit der On-Premises-Variante von Exchange beschäftigen, sind die aufgeführten PowerShell-Cmdlets auch für Exchange Online verfügbar. Damit ist gewährleistet, dass einmal erstellte One-Liner und Skripte in der Regel sowohl lokal als auch in Hybrid- und Cloudumgebungen in gleicher Weise einsetzbar sind.

Verbindung zum Server herstellen

Damit Sie mit Hilfe der PowerShell Befehle an den Exchange-Server senden können, müssen Sie zunächst eine Verbindung zu diesem herstellen.

In unserem Szenario gehen wir davon aus, dass der Exchange-Administrator sich nicht direkt etwa per RDP auf dem Exchange-Server anmeldet, sondern auf ei-

nem Windows-Client arbeitet. Darüber hinaus soll die Installation zusätzlicher PowerShell-Module auf dem Client vermieden werden. Um das zu erreichen, machen Sie sich das implizite Remoting der PowerShell zunutze. Sie greifen damit direkt auf den Exchange-Server zu und führen dort die erforderlichen Cmdlets aus.

Zunächst verwenden Sie das Cmdlet "New-PSSession", um eine Verbindung zum Server "Exchange.company.net" aufzubauen. Für die erfolgreiche Ausführung dieser Cmdlets benötigen Sie ein Benutzerkonto, das die erforderlichen Rechte auf diesem Exchange-Server besitzt. Die Anmeldeinformationen übergeben Sie dem Cmdlet "Get-Credentials".

Im letzten Schritt importieren Sie dann mit der Ausführung des "Import-PSSession"-Befehls die Exchange-Cmdlets in unsere lokale PowerShell-Umgebung. Somit können Sie aus unserer aktuellen Session direkt auf den Exchange-Server zugreifen. Hier ein Beispiel für den gesamten Vorgang:

```
$MyCred = Get-Credential
```

```
$Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri http://Exchange.compa-
```

```
ny.net/PowerShell/ -Authentication Kerberos -Credential $MyCred
```

```
Import-PSSession $Session -DisableNameChecking
```

Ein erster Test mit dem Cmdlet "Get-Mailbox" zeigt die Liste aller Exchange-Postfächer. Somit ist die Verbindung zum Server erfolgreich hergestellt und wir können uns den Abwesenheitseinstellungen widmen.

Bestehende Einstellungen abfragen

Für die Abfrage der aktuellen Einstellungen steht das Cmdlet "Get-MailboxAutoReplyConfiguration" zur Verfügung. Damit zeigen Sie beispielsweise Angaben einzelner Benutzer, aller Benutzer oder der Benutzer einer bestimmten Gruppe oder OU an. Um die Konfiguration eines einzelnen Benutzers zu prüfen, nutzen Sie den Parameter "-Identity". Für die eindeutige Identifizierung eines Users können Sie hier beispielsweise Name, Alias, Distinguished Name oder die E-Mail-Adresse verwenden, etwa so:

```
Get-MailboxAutoReplyConfiguration -Identity thomas@company.net
```

```
Get-MailboxAutoReplyConfiguration -Identity Thomas
```

```
PS C:\test\WebService> Get-MailboxAutoReplyConfiguration -id Thomas@company.net
RunspaceId                : 7482a3fb-bd42-4fd0-887a-d35cdc47a119
AutoDeclineFutureRequestsWhenOOOF : False
AutoReplyState             : Scheduled
CreateOOFEvent             : False
DeclineAllEventsForScheduledOOOF : False
DeclineEventsForScheduledOOOF : False
EventsToDeleteIDs         :
EndTime                    : 02/08/2019 00:00:00
ExternalAudience          : All
ExternalMessage            : <html>
                             <body>
                             I'm out of the office from 7/29/2019 until 8/2/2019
                             </body>
                             </html>
InternalMessage            : <html>
                             <body>
                             I'm out of the office from 7/29/2019 until 8/2/2019
                             </body>
                             </html>
DeclineMeetingMessage      :
OOFEEventSubject          :
StartTime                  : 29/07/2019 00:00:00
MailboxOwnerId             :
Identity                   : company.net/APAC/Singapore/Consulting/Thomas
IsValid                    : True
ObjectState                 : Unchanged
```

Bild 1: So sehen die mit der PowerShell abgefragten Abwesenheitsinformationen für einen einzelnen Benutzer aus.

Als Ergebnis erhalten Sie den aktuellen Status und sehen in unserem Beispiel, dass für diesen Benutzer eine Abwesenheit (AutoReplyState: Scheduled) vom "29/07/2019 (StartTime)" bis "02/08/2019 (EndTime)" konfiguriert ist. Darüber hinaus ist erkennbar, dass der Text der Abwesenheitsnachricht für interne Empfänger (InternalMessage) und externe Empfänger (ExternalMessage) gesetzt ist.

Schauen wir uns nun an, wie Sie Abwesenheitseinstellungen für mehrere Benutzer abfragen können. Hierfür bedienen Sie sich der PowerShell-Pipeline (|). Mit ihrer Hilfe lassen sich einzelne Befehle einfach miteinander verketteten. Das gibt uns eine Vielzahl von Möglichkeiten bei der Ausführung von PowerShell-Befehlen und Skripten.

Ist die Exchange-Umgebung von der Größe her überschaubar, können Sie sehr einfach die Out-of-Office-Einstellungen aller Benutzer anzeigen. Dazu rufen Sie folgendes Kommando auf:

```
Get-Mailbox | Get-MailboxAutoReplyConfiguration
```

Als Ergebnis erhalten Sie eine komplette Liste aller Abwesenheitskonfigurationen. In größeren Umgebungen kann diese Liste natürlich sehr lang und unübersichtlich ausfallen. Wenn Sie nur die Benutzer interessieren, die derzeit eine Abwesenheit

terminiert haben, erweitern Sie den obigen Befehl um "Where-Object". Damit schränken Sie die Ausgabe auf die Postfächer ein, bei denen der "AutoReplyState" auf "Scheduled" gesetzt ist.

```
Get-Mailbox | Get-MailboxAutoReplyConfiguration | Where-Object {
    $_.AutoReplyState -eq "scheduled"
}
```

Um gezielt die aktiven Abwesenheiten ohne Endzeitpunkt zu sehen, ergänzen Sie "Where-Object" um "-OR \$_.AutoReplyState -eq "enabled". Diese Konfiguration findet sich in der Regel bei Postfächern ausgeschiedener Mitarbeiter. Damit senden Sie Geschäftspartnern eine Benachrichtigung über den Weggang des Mitarbeiters mit der gleichzeitigen Information über die Kontaktdaten des Nachfolgers. Im Ergebnis sieht unser Kommando nun wie folgt aus:

```
Get-Mailbox | Get-MailboxAutoReplyConfiguration | Where-Object {
    $_.AutoReplyState -eq "scheduled"
    -OR $_.AutoReplyState -eq "enabled"
}
```

Wollen Sie nur die Namen der Benutzer wissen, die eine Abwesenheit aktiviert haben, hilft das Hinzufügen von "fl identity":

```
Get-Mailbox | Get-MailboxAutoReplyConfiguration | Where-Object {
```

```
    $_.AutoReplyState -eq "scheduled"
    -OR $_.AutoReplyState -eq "enabled"
} | fl identity
```

Damit sich diese Informationen zum Beispiel für Dokumentationszwecke weiterverwenden lassen, schreiben Sie die Liste mit "Out-file -filepath C:\test\result.txt" in eine Textdatei:

```
Get-Mailbox | Get-MailboxAutoReplyConfiguration | Where-Object {
    $_.AutoReplyState -eq "scheduled"
    -OR $_.AutoReplyState -eq "enabled"
} | fl identity | Out-file -filepath C:\test\result.txt
```

Wiederkehrende Abfragen durchführen

Unser Beispiel eignet sich natürlich auch für einen geplanten Task. Die Liste ließe sich beispielsweise einmal wöchentlich erstellen und auf einer Dateifreigabe bereitstellen. Eine weitere Möglichkeit besteht darin, eine E-Mail mit diesem Report regelmäßig an das Personalteam zu senden. Dazu verwenden Sie das Cmdlet "Send-MailMessage". Im folgenden Beispiel versenden wir eine E-Mail mit der Datei "result.txt" und der Absender erhält eine Zustellbestätigung:

```
Send-MailMessage -From 'Admin <administrator@company.net>' -To 'Theo <theo@company.net>' -Subject 'Weekly report: Out-of-Office' -Body "Please find the list attached" -Attachments C:\test\result.txt -Priority High -DeliveryNotificationOption OnSuccess, OnFailure -SmtpServer 'exchange.company.net'
```

Um die Bildschirmausgabe größerer Datenmengen einfacher lesbar zu machen, stellt die PowerShell ebenfalls Möglichkeiten zur Verfügung. So können Sie mit "Out-GridView" eine interaktive Tabelle der Abwesenheitsinformationen erzeugen und darin filtern und sortieren. Nehmen Sie unser Anfangsbeispiel *Get-Mailbox | Get-MailboxAutoReplyConfiguration* und ergänzen es mit "| Out-GridView", erhalten Sie ein Ergebnis wie in Bild 2. Natürlich haben Sie auch die Möglichkeit festzulegen, welche Informationen die Tabelle anzeigen soll. Hierfür verwenden Sie "Se-

| eteIds | EndTime | ExternalAudience | ExternalMessage | InternalMessage | MailboxOwnerId |
|--------|---------------------|------------------|---|--|--------------------------------------|
| | 07/07/2019 15:00:00 | All | <html><body>I'm out of the office from 2/12/2019 until 2/19/2019</body></html> | <html><body>I'm out of the office from 2/12/2019 until 2/19/2019</body></html> | company.net/Users/Administrator |
| | 07/07/2019 15:00:00 | All | <html><body>I'm out of the office from 5/2/2019 until 5/30/2019. Your mail will not be forwarded.</body></html> | <html><body>I'm out of the office from 5/2/2019 until 5/30/2019. Your...</body></html> | company.net/NA/US/IT/Adrian |
| | 07/07/2019 15:00:00 | All | <html><body>I'm out of the office from 6/5/2019 until 6/13/2019. Your mail will not be forwarded.</body></html> | <html><body>I'm out of the office from 6/6/2019 until 6/13/2019. Your...</body></html> | company.net/EMEA/UK/Consulting/Alice |
| | 07/07/2019 15:00:00 | All | <html><body>I'm out of the office from 3/29/2019 until 3/21/2019</body></html> | <html><body>I'm out of the office from 3/29/2019 until 3/21/2019</body></html> | company.net/NA/US/Sales/Anna |
| | 07/07/2019 15:00:00 | All | <html><body>I'm out of the office from 6/21/2019 until 7/5/2019. Your mail will not be forwarded.</body></html> | <html><body>I'm out of the office from 6/21/2019 until 7/5/2019. Your...</body></html> | company.net/EMEA/UK/Sales/David |

Bild 2: Die Liste der Abwesenheitseinstellungen als interaktive Tabelle.

lect-Object" und geben die Namen der gewünschten Felder an:

```
Get-Mailbox | Get-MailboxAutoReply-
Configuration | Select-Object
Identity,Starttime,Endtime,Exter-
nalMessage,InternalMessage |
Out-GridView
```

Erstellen von Abwesenheitskonfigurationen

Nachdem Sie bereits einige Möglichkeiten zum Anzeigen vorhandener Abwesenheitseinstellungen kennengelernt haben, geht es nun um die Bearbeitungsmöglichkeiten dieser Einstellungen. Dazu schauen wir uns das Cmdlet "Set-MailboxAutoReplyConfiguration" etwas näher an.

Beginnen wir mit dem Erstellen einer Abwesenheitsbenachrichtigung für einen User. Nehmen wir an, dieser User hat das Unternehmen verlassen und bei E-Mails an dieses Postfach sollen dauerhaft Benachrichtigungen versendet werden. Der Benachrichtigungstext soll dabei für interne und externe E-Mails identisch sein. Der entsprechende Befehl würde dann zum Beispiel so aussehen:

```
Set-MailboxAutoReplyConfiguration
-Identity Thomas@company.net
-AutoReplyState Enabled
-InternalMessage "This user is no
longer working for us. Please
```

```
write to Sandra@company.net"
-ExternalMessage "This user is no
longer working for us. Please
write to Sandra@company.net"
```

In einem zweiten Beispiel ist ein Mitarbeiter krank geworden und kann die entsprechende Abwesenheit nicht selbst erstellen, also übernimmt das die IT-Verwaltung. Auf Basis der Krankmeldung ist nun auch ein Endzeitpunkt für die Abwesenheit bekannt, und die Out-of-Office-Konfiguration lässt sich demgemäß so einrichten:

```
Set-MailboxAutoReplyConfiguration -
Identity Thomas@company.net -Auto-
ReplyState Scheduled
-InternalMessage "I'm on sick leave
until 12/07/2019. Your email will
not be forwarded." -ExternalMessa-
ge "I'm on sick leave until
12/07/2019. Your email will not be
forwarded." -EndTime "12/07/2019
00:00:00"
```

Die Mitteilungen für interne und externe Empfänger können Sie im Bedarfsfall auch unterschiedlich konfigurieren.

Für unser drittes Beispiel wollen wir Abwesenheitsmitteilungen für mehrerer Benutzer erstellen. Ein Szenario hierfür wäre beispielsweise eine Schulung für ein gemeinsames Team oder eine Unternehmens-

veranstaltung. Für die Umsetzung benötigen wir zunächst eine Datei, die die zu verarbeitenden Benutzer enthält. Zum Erstellen nutzen Sie zum Beispiel folgenden Befehl:

```
Get-Mailbox |Select-Object Prima-
rySmtpAddress | ConvertTo-CSV
-NoTypeInfo | Select-Object
-Skip 1 | %{$_ -replace '"','"} |
out-file C:\test\user.txt
```

Wir erhalten die primären E-Mail-Adressen aller Mailboxen als Textdatei. Mit "Select-Object -Skip 1" eliminieren wir die Überschrift und mit "%{\$_ -replace '"','"}" werden die Anführungszeichen entfernt.

Die Liste können Sie nun über eine ForEach-Schleife verarbeiten. Dabei erstellen Sie für jeden Eintrag in der Textdatei (sprich: Postfach) die festgelegte Abwesenheitskonfiguration:

```
$Users = Get-Content
C:\test\user.txt

$(foreach ($User in $Users)
{Set-MailboxAutoReplyConfiguration
$User -AutoReplyState Scheduled
-StartTime "08/09/2019" -EndTime
"10/09/2019" -ExternalMessage "At
the moment I'm attending a train-
ing event. I will answer your
```

```

ExternalMessage : <html>
                  <head>
                  <style type="text/css" style="display:none">
                  <!--
                  p
                  {margin-top:0;
                  margin-bottom:0}
                  -->
                  </style>
                  </head>
                  <body dir="ltr">
                  <div id="divtagdefaultwrapper" dir="ltr" style="font-size:12pt; color:#000000; font-family:Calibri,Helvetica,sans-serif">
                  <span style="color:rgb(255,0,0)">I'm on tour right now. <br>
                  </span>Your email will not be forwarded.
                  <div>Visit our website <a href="http://www.scriptrunner.com" class="OWAAutoLink">
                  www.scriptrunner.com</a></div>
                  <div><br>
                  </div>
                  </div>
                  </body>
                  </html>

```

Bild 3: Auch HTML-Benachrichtigungen lassen sich mit den entsprechenden Vorlagen konfigurieren.

```

email asap." -InternalMessage "At
the moment I'm attending a train-
ing event. I will answer your
email asap."
})

```

HTML nutzen

Bisher haben wir die Benachrichtigungsmitteilungen als reine Textnachricht verfasst, aber auch die Verwendung von HTML ist selbstverständlich möglich.

In diesem Fall ist der HTML-Nachrichtentext zunächst in einer Datei zu speichern. Die Inhalte dieser Datei werden dann bei der Ausführung von "Set-MailboxAutoReplyConfiguration" an das Cmdlet übergeben.

In Schritt 1 weisen Sie den Inhalt der vorgefertigten Nachricht der Variable "\$message" zu:

```

$message = get-content
C:\test\html_oof.txt

```

Danach konfigurieren Sie "\$message" bei externen und internen Nachrichten:

```

Set-MailboxAutoReplyConfiguration
-Identity Nora -AutoReplyState
Scheduled -StartTime "08/09/2019"
-EndTime "10/09/2019" -External-
Message $message -InternalMessage
$message

```

Führen Sie nun "Get-MailboxAutoReplyConfiguration" für diesen Benutzer aus, sehen Sie ein Ergebnis wie in Bild 3. Ist die Durchführung der gewünschten Aufgaben beendet, schließen Sie mit *Remove-PSSession -ID n* die aktuelle Session. Die ID der Session erhalten Sie mit *Get-PSSession*.

Möglichkeiten zur Delegation eruieren

Sie haben nun Wege kennengelernt, wie Administratoren Abwesenheitseinstellungen anzeigen und erstellen können. Was aber, wenn diese Aufgaben an Helpdesk-Mitarbeiter oder Benutzer in Fachbereichen wie beispielsweise der Personalabteilung delegiert werden sollen? Folgende Fragen sind dabei zunächst zu beleuchten:

1. Können die Mitarbeiter mit der PowerShell umgehen?
2. Dürfen/können Sie diesen Mitarbeitern Administrationsberechtigungen für die Bearbeitung von Postfächern vergeben?

Lautet die Antwort auf beide Fragen "Ja", können Sie die oben aufgezeigten Möglichkeiten problemlos in dieser Form delegieren. Sollte das nicht der Fall sein, weil die Kollegen zum Beispiel kein PowerShell-Know-how besitzen, oder aber wenn die Vergabe der Berechtigungen aus organisatorischen Gründen nicht möglich ist, müssen Sie sich spezielle Werkzeuge zu Nutze machen. Diese sollten folgende Eigenschaften mitbringen:

1. Zentrale Ausführung und Überwachung aller PowerShell-Skripte
2. Zentrale und sichere Speicherung der erforderlichen Credentials zur Ausführung der PowerShell
3. Ausführung von PowerShell-Skripten auch für Nicht-PowerShell-Experten

Diese Punkte sind beispielsweise mit der Software ScriptRunner sehr einfach umsetzbar. Hierbei werden alle PowerShell-Skripte zentral gespeichert, verwaltet und ausgeführt.

Darüber hinaus ermöglicht dieser Ansatz auch das gemeinsame Entwickeln von

PowerShell-Skripten im Team. Die benötigten Anmelde-Informationen sind im Windows Credential Store oder einem Passwortserver sicher gespeichert. Der Benutzer, der eine Abwesenheitskonfiguration erstellen soll, greift mit einer komfortablen Weboberfläche auf die Funktionen zu und hat dabei keinerlei Berührungspunkte mit der darunterliegenden PowerShell-Welt.

Die Oberfläche selbst wird im Übrigen automatisch aus dem jeweiligen PowerShell-Skript erstellt. Damit entfällt zudem das manuelle Programmieren von GUI-Code komplett. Ein weiterer Vorteil ergibt sich aus der Tatsache, dass in der Abwesenheitsbenachrichtigung Variablen für Start und Ende der Abwesenheit zur Verfügung stehen.

Diese Variablen werden automatisch aus den Datumsangaben übernommen. Das vereinfacht das Erstellen einer Out-of-Office-Konfiguration zusätzlich und sorgt außerdem dafür, dass zukünftig die Benachrichtigungen immer mit korrekten und aktuellen Datumsinformationen das Haus verlassen.

Fazit

Die PowerShell ist ein hervorragendes Werkzeug für die einfache Umsetzung wiederkehrender Aufgaben bei der Exchange-Administration. Mit entsprechenden Erweiterungen ist es aber auch eine perfekte Basis für die sichere und komfortable Delegation festgelegter Anwendungsfälle an Mitarbeiter und Teams außerhalb der IT-Administration. (In) 

Heiko Brenn ist Head of International Business bei der ScriptRunner Software GmbH.