Office 365 Exchange Online Administration using PowerShell

Kamal Abburi
Powershell ISE (Integrated Scripting environment)
```powershell
Get-Process
```
There are two PowerShell interfaces that can be used to connect to and manage Microsoft Office 365:

2. Remote PowerShell for Exchange Online.

PowerShell:
- Uses HTTPS to connect securely to the DataCenter
- Does not require Exchange Server management tools
- Requires Windows Management Framework, which contains Windows PowerShell v2 and WinRM 2.0 (installed by default on Windows 7 and Windows 2008 R2)
• Azure Active Directory Module and Office 365 portal.
• http://aka.ms/aadposh
• Connecting to Office 365
• `Import-Module MSOnline`
• `Connect-MsolService`
• `Get-Command | Where {$_ModuleName -match "MSOnline"}`
Remote PowerShell for Exchange Online

- Administrators do not need to install any Exchange Server management tools in order to use remote Windows PowerShell for Exchange Online
- To connect to Exchange Online via remote PowerShell
  1. Open **Windows PowerShell**.
  2. Save the Exchange Online administrator credentials as a variable
     - `$Cred = Get-Credential`
  3. Create a new session using the saved username and password
  4. Import the session:
     - `Import-PSSession $Session`
  5. To finish the session, remove it by typing:
     - `Remove-PSSession $Session`
Connect to all Office 365 services in a single Windows PowerShell window

- Set-ExecutionPolicy RemoteSigned
- $credential = Get-Credential

• Import-Module MsOnline
• Connect-MsolService -Credential $credential

• Import-Module Microsoft.Online.SharePoint.PowerShell -DisableNameChecking
• Connect-SPOService -Url https://domainhost-admin.sharepoint.com -credential $credential

• Import-Module SkypeOnlineConnector
• $sfboSession = New-CsOnlineSession -Credential $credential
• Import-PSSession $sfboSession

• Import-PSSession $exchangeSession -DisableNameChecking
RBAC and the Shell

Source: TechNet
cmdlets

- Get-Command
- Get-ExCommad
- Get-Command *mailbox*
- Get-Command –Noun Mailbox
- Get-Command –Verb Restore

Tip: Start-Transcript
The Power of TAB

PowerShell auto completion

- When entering cmdlets or parameters, use the <TAB> key to auto complete.
- The <TAB> key can also be used to scroll through parameters or cmdlets.
- Try typing `get-mailbox -<TAB>`
  - Continue pressing <TAB> to scroll through all parameters.
- Try typing `get-m<TAB>`
  - Continue pressing <TAB> to scroll through all cmdlets that start with “get-a”
Get-Help

Use help to find cmdlets and categories

- Get-help *User*
- Get-help -role *UM*
- Get-help –component *recipient*

Use help pages to drill into more detail

- Get-mailbox -? | more
- Help set-mailbox
- Get-help get-mailbox –Detailed
- Get-help set-mailbox –Examples
- Get-Help <cmdlet> -Parameter <parameter name>
Alias

PowerShell Shorthand Notation

- Aliases are used to shorten common commands in PowerShell.
- Use `get-alias` to see all aliases
- Create your own alias using `new-alias`

<table>
<thead>
<tr>
<th>Alias</th>
<th>Cmdlet</th>
</tr>
</thead>
<tbody>
<tr>
<td>dir</td>
<td>get-childitem</td>
</tr>
<tr>
<td>cd</td>
<td>set-location</td>
</tr>
<tr>
<td>rm</td>
<td>remove-item</td>
</tr>
<tr>
<td>rmdir</td>
<td>remove-item</td>
</tr>
<tr>
<td>copy</td>
<td>copy-item</td>
</tr>
<tr>
<td>echo</td>
<td>write-output</td>
</tr>
<tr>
<td>del</td>
<td>remove-item</td>
</tr>
<tr>
<td>move</td>
<td>move-item</td>
</tr>
</tbody>
</table>
Parameters

- Provide information to the cmdlet
- Control how the cmdlet performs its task
- Verb-Noun -ParameterName <ParameterValue>

Types
- Positional (Identity)
- Named (Specify the Parameter)
- Boolean($true, $false)
- Switch(confirm)
- Common(Verbose, Debug, ErrorAction)
Syntax

- **Verb-Noun -ParameterName <ParameterValue>**
- hyphen indicates a parameter
- Space in Parameter Value: Double quotation marks (""
- **Single Quote vs Double Quote**
  - "$Server Example"
  - ‘$Server Example’
- **Escape Character**
  - “Gold is `\$1600"
Exploring Parameters

Explore the properties of output objects using format-list

- Get-Mailbox TestUser | format-list
- Get-Mailbox TestUser | fl *quota*
- Get-Mailbox TestUser | fl *
- Get-ExchangeServer -Status | fl *
- Get-Mailbox | FT Name,Database

Tab it Set-Mailbox-<tab>
Operators

= value on the right side of the equal sign is assigned to the variable on the left side

! logical **NOT** operator. How do I say “Not Equal To”

>, >> send the output of a command to a file

{} – Expression

$ Variable

+, -, *, %

-eq, -ne, -Like, -and, -or, -gt, -lt


Tip: Tee-Object
Pipeline

Why pipe between cmdlets?

- You don’t have to. You can use set- cmdlets to directly modify an object.
  - Set-mailbox jack –issuewarningquota 90MB
- However, set- cmdlets only modify one object at a time. They cannot be wildcarded.
- Piping between cmdlets allows you to modify multiple objects with one command.
- PowerShell’s ability to pipe information from cmdlet to cmdlet makes bulk administration simple.
Displaying

Methods for displaying information

The default information displayed by a cmdlet contains only a subset of the available information.

Use these cmdlets to display more information:

• Format-list (alias FL)
  – Returns object properties in list form
  – * returns all properties, specific properties can be returned by distinct or wildcarded name

• Format-table (alias FT)
  – The same concept of format-list, but information is displayed in a table view.

• Sort-object (alias sort)
  – Sort information in a variety of ways based on object properties

  Get-mailboxstatistics | sort –property itemcount –desc

• Group-object (alias group)
  – Groups information based on a common object property

  Get-mailbox |group –property Database
Import

Get-Content

- If the content you want to import is not structured, use get-content.
- This can be useful when manipulating log files or other data not stored in .csv form.

Import-Csv

- Any information stored in a comma separated value format can be imported using import-csv.
- The first row of the import file defines the properties for the imported object.
- All other rows are objects defined according to the previously imported properties.
- Once the information is in object form, it can take advantage of other Powershell cmdlets