Office 365 PowerShell Guide

21 February 2018 10:00

All actions that can be performed in the Office 365 web interface can be done (and automated) through PowerShell. This guide is to list some of the common actions performed during Office 365 validation and explain the commands that allow you to execute them through PowerShell.

Create Office 365 Trial

This is the one step that cannot be automated via PowerShell. Follow the steps in Setup Office 365 Trial to setup a trial Office 365 account and record the Administrator username and password for use later.

Explanation of the different connections

Office 365 is made up of a number of different Microsoft online services: e.g. Exchange Online and SharePoint Online. These individual services are cloud versions of Microsofts on-premise products that in many cases pre-date the introduction of Office 365. Office 365 is essentially an Azure Active Directory directory that oversees the users and licensing for the individual online service. This is why the individual services have different sets of PowerShell commands and different methods of connecting.

For the purposes of this guide we are interested in connections to:

- Office 365 - the Azure AD service that is managing our trial company
- Exchange Online - the instance of Exchange Online that is providing an email service for this account

Commands given below will require a connection to be established to either one or both of these services and will specify which. Where a section is marked O365 you must have an Office 365 connection established in your PowerShell session, where it is marked EOL you must have an Exchange Online connection.

You can have a script with both connections open simultaneously, you can even have more connections to other services (see https://docs.microsoft.com/en-us/office365/enterprise/powershell/connect-to-all-office-365-services-in-a-single-windows-powershell-window). However note the limitation on connecting to Exchange Online from within the Veritas COMMUNITY domain mentioned below.

Executing PowerShell commands

The best tool for executing PowerShell commands interactively is Visual Studio Code with the PowerShell extension installed https://marketplace.visualstudio.com/items?itemName=ms-vscode.PowerShell This has replaced PowerShell ISE as the recommended editor for PowerShell scripts. To execute a PowerShell command once this extension has been installed create a file with a ps1 extension, open it with Visual Studio code and you can just select the command and press F8.

On a fresh install of Windows it is normally necessary to relax the execution policy for PowerShell scripts to RemoteSigned. This can be done with the following command:

```
Set-ExecutionPolicy RemoteSigned
```

Connecting to Office 365

An Office 365 connection is required for any of the scripts marked O365 below.

Pre-requisites

On a new Windows install you will need to install the Azure Active Directory Module for Windows PowerShell. Many of the instructions online for installing this are outdated. On an up to date Windows 10 computer you can install this with PowerShellGet by running the following commands. These only need to be run once to install the components.

```
Set-PSRepository -Name PSGallery -InstallationPolicy Trusted

Install-Module MsOnline
```

Note for Windows Server 2012 R2

The above will require PowerShell v5, to install on Windows Server 2012 R2 install Windows Management Framework 5.0 from https://www.microsoft.com/en-us/download/details.aspx?id=50395 making sure to chose Win8.1AndW2K12R2-KB3134758-x64.msu

Then run the following sequence of commands:

```
Install-PackageProvider -Name NuGet
Set-PSRepository -Name PSGallery -InstallationPolicy Trusted
Install-Module MsOnline -Force
Import-Module MsOnline
```

Note for systems that used the Windows Azure Active Directory Module for Windows

Before PowerShell 5 modules were installed via msi packages and installed to a different location. If using an older module you will see warnings like the following:

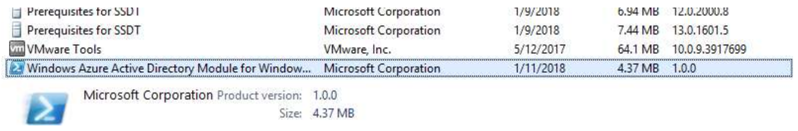
WARNING: There is a newer version of the Microsoft Online Services Module. Your current version will still work as expected, however the latest version can be downloaded at https://portal.microsoftonline.com.

The new module can be installed using the commands given above (Install-Module), but for it to be used the older modules must be uninstalled. To determine if you have the older modules run the following command:

```
(get-item C:\Windows\System32\WindowsPowerShell\v1.0\Modules\MSOnline\Microsoft.Online.Administration.Automation.PSModule.dll).VersionInfo.FileVersion
```

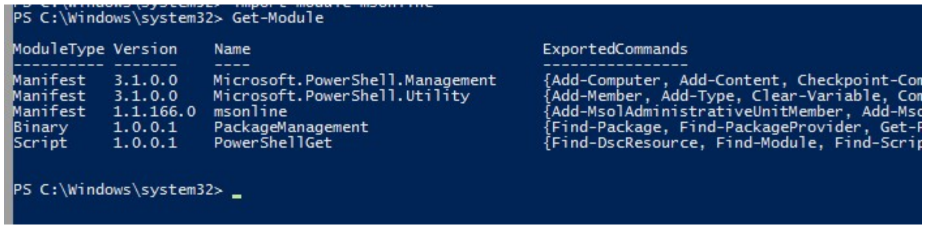
If they are present uninstall them through the Add/Remove programs dialog, choosing the **Windows Azure Active Directory Module for Windows**:

```
Prerequisites for SSDT          Microsoft Corporation     1/9/2018     6.94 MB    12.0.2000.8
```



```
Prerequisites for SSDT                              Microsoft Corporation   1/9/2018    6.94 MB   12.0.2000.8
Prerequisites for SSDT                              Microsoft Corporation   1/9/2018    7.44 MB   13.0.1601.5
VMware Tools                                        VMware, Inc.            5/12/2017   64.1 MB   10.0.9.3917699
Windows Azure Active Directory Module for Window... Microsoft Corporation   1/11/2018   4.37 MB   1.0.0
Microsoft Corporation Product version: 1.0.0
                               Size: 4.37 MB
```

Run Get-Module to confirm that a newer version of MsOnline module is being used, should be >= 1.1.166.0 and only one version should be present



```
PS C:\Windows\system32> Get-Module

ModuleType Version    Name                                ExportedCommands
---------- -------    ----                                ----------------
Manifest   3.1.0.0    Microsoft.PowerShell.Management     {Add-Computer, Add-Content, Checkpoint-Com
Manifest   3.1.0.0    Microsoft.PowerShell.Utility        {Add-Member, Add-Type, Clear-Variable, Com
Manifest   1.1.166.0  msonline                            {Add-MsolAdministrativeUnitMember, Add-Msc
Binary     1.0.0.1    PackageManagement                   {Find-Package, Find-PackageProvider, Get-F
Script     1.0.0.1    PowerShellGet                       {Find-DscResource, Find-Module, Find-Scrip

PS C:\Windows\system32> _
```

Connecting

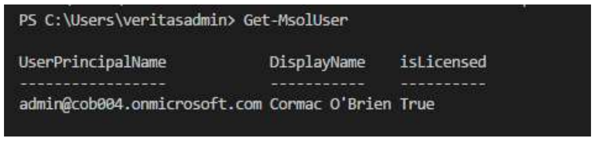
Connecting to Office 365 uses a modern set of PowerShell cmdlets that are easy to install and use. Once the pre-requisities above are done once all that is required to connect to O365 is the following:

```
$username = "admin@cob004.onmicrosoft.com"
$password = ConvertTo-SecureString "W3lcome!W3lcome!" -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential $username, $password

Connect-Msolservice -Credential $cred
```

Verify the connection by running the following command that should list the current O365 users:

```
Get-MsolUser
```



```
PS C:\Users\veritasadmin> Get-MsolUser

UserPrincipalName            DisplayName    isLicensed
-----------------            -----------    ----------
admin@cob004.onmicrosoft.com Cormac O'Brien True
```

The storing of credentials in the manner above would normally be frowned upon, but as they are not sensitive credentials we can do this as a convenience. To keep sensitive credentials safe you would instead prompt the user for them by using the following command:

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

Connecting to Exchange Online

An Exchange Online connection is required for any of the scripts marked EOL below. Connecting to Exchange online is very different from connecting to Office 365. Why? Probably because it predates Office 365. Think of it as opening a remote connection to a giant Exchange Server in the cloud. That is what the New-PSSession and Import-PSSession commands are doing. The Import-PSSession command also copies down all the cmdlets required into the location PowerShell session, this can take a while to complete.

The most important thing to realise is that you can **NOT** connect to EOL from a Veritas computer within the COMMUNITY domain. This is because Basic Authentication is for the connection to the remote Exchange server and this is blocked within the COMMUNITY domain. You must use a sandbox or any other computer not connected to the domain to run the EOL powershell commands. The Office 365 connection does not have this limitation.

Pre-requisites

Instructions online will say it is necessary to install the 64-bit version of the Microsoft Online Services Sign-in Assistant, but I've not found this to be the case when connected to the remote server using PSSession commands.

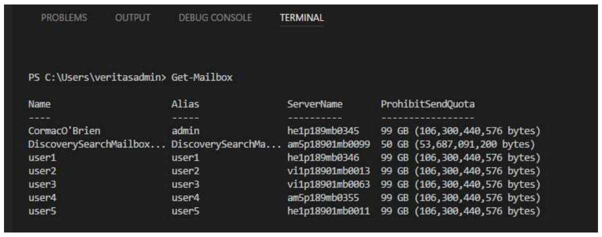
Connecting

```
$username = "admin@cob004.onmicrosoft.com"
$password = ConvertTo-SecureString "W3lcome!W3lcome!" -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential $username, $password
$session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri https://outlook.office365.com/powershell-liveid/ -Credential $cred -
Authentication Basic -AllowRedirection
Import-PSSession $session
```

Test the connection by running the following command

```
Get-Mailbox
```

It should show something like the following:



```
PS C:\Users\veritasadmin> Get-Mailbox

Name                      Alias                ServerName         ProhibitSendQuota
----                      -----                ----------         -----------------
CormacO'Brien             admin                he1p189mb0345      99 GB (106,300,440,576 bytes)
DiscoverySearchMailbox... DiscoverySearchMa... am5p18901mb0099    50 GB (53,687,091,200 bytes)
user1                     user1                he1p189mb0346      99 GB (106,300,440,576 bytes)
user2                     user2                vi1p18901mb0013    99 GB (106,300,440,576 bytes)
user3                     user3                vi1p18901mb0063    99 GB (106,300,440,576 bytes)
user4                     user4                am5p189mb0355      99 GB (106,300,440,576 bytes)
user5                     user5                he1p18901mb0011    99 GB (106,300,440,576 bytes)
```

Creating Office 365 Users (O365)

The first task will normally be to create a couple of user accounts in Office 365 and assign them licenses that will create services in Exchange Online. This can be achieved easily with the following PowerShell:

```
$sku = Get-MsolAccountSku

$userPassword = "W3lcome!W3lcome!"

New-MsolUser -DisplayName "John User1" -FirstName John -LastName User1 -UserPrincipalName user1@cob004.onmicrosoft.com -ForceChangePassword $false -Password $userPassword -PasswordNeverExpires $true -
UsageLocation IE -LicenseAssignment $sku.AccountSkuId
```

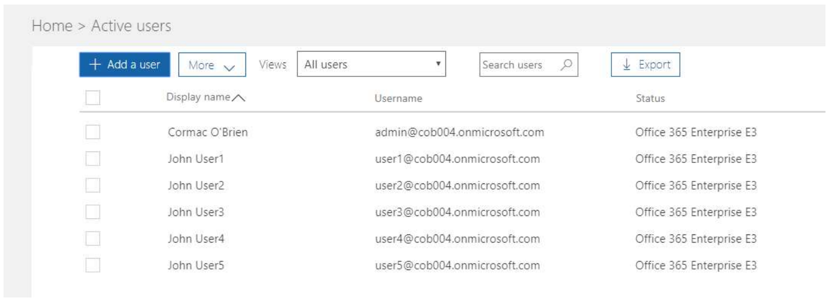
The advantage of PowerShell is that we can very easily turn this into a script to create multiple users. For example to create 5 user accounts for a new Office 365 company named cob004 you can use the following:

```
$sku = Get-MsolAccountSku
$userPassword = "W3lcome!W3lcome!"

For ($i=1; $i -lt 6; $i++)
{
    $companyName = "cob004"
    $firstName = "John"
    $lastName = "User$i"
    $displayName = "$firstName $lastName"
    $userPrincipalName = "user$i@$companyName.onmicrosoft.com"

    New-MsolUser -DisplayName $displayName -FirstName $firstName -LastName $lastName -UserPrincipalName $userPrincipalName `
            -ForceChangePassword $false -Password $userPassword -PasswordNeverExpires $true -UsageLocation IE -LicenseAssignment $sku.AccountSkuId
}
```

This results in 5 users with the same password and with licenses assigned:



Display name	Username	Status
Cormac O'Brien	admin@cob004.onmicrosoft.com	Office 365 Enterprise E3
John User1	user1@cob004.onmicrosoft.com	Office 365 Enterprise E3
John User2	user2@cob004.onmicrosoft.com	Office 365 Enterprise E3
John User3	user3@cob004.onmicrosoft.com	Office 365 Enterprise E3
John User4	user4@cob004.onmicrosoft.com	Office 365 Enterprise E3
John User5	user5@cob004.onmicrosoft.com	Office 365 Enterprise E3

Change a user's password (O365)

```
$userPassword = "W3lcome!W3lcome!"
Set-MsolUserPassword -UserPrincipalName user3@cob004.onmicrosoft.com -ForceChangePassword $false -NewPassword $userPassword
```

Block (disable) and unblock a user (O365)

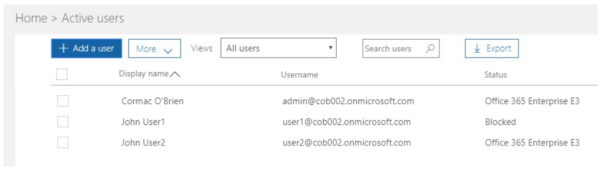
Some of the Office 365 tests call for blocking and unblocking a user. This is achieved with the followed:

```
Set-MsolUser -UserPrincipalName user3@cob004.onmicrosoft.com -BlockCredential $true
```

and unblock with

```
Set-MsolUser -UserPrincipalName user3@cob004.onmicrosoft.com -BlockCredential $false
```

Blocked users will appear like this:



Display name	Username	Status
Cormac O'Brien	admin@cob002.onmicrosoft.com	Office 365 Enterprise E3
John User1	user1@cob002.onmicrosoft.com	Blocked
John User2	user2@cob002.onmicrosoft.com	Office 365 Enterprise E3

Delete and restore user (O365)

The following commands will delete a user, show a list of deleted users and then restore the user:

```
Remove-MsolUser -UserPrincipalName user3@cob004.onmicrosoft.com -Force
```

```
Get-MsolUser -All -ReturnDeletedUsers
```

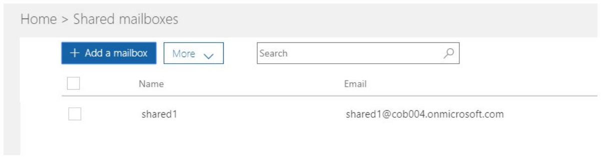
```
Restore-MsolUser -UserPrincipalName user3@cob004.onmicrosoft.com
```

Create a shared mailbox (EOL)

Execute the following:

```
New-Mailbox -Name shared1 -Shared
```

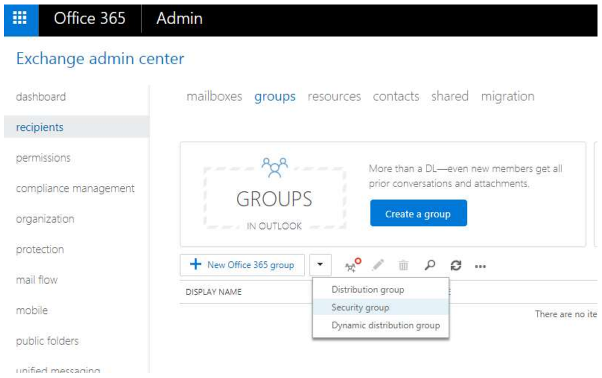
This will create a shared mailbox with the email address like shared1@cob004.onmicrosoft.com:



Name	Email
shared1	shared1@cob004.onmicrosoft.com

Create a mail enabled security group (EOL)

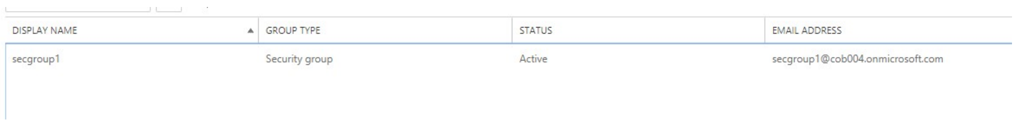
There are multiple types of groups within Office 365 and Exchange Online. Some appear to be deprecated in favour of new Office 365 Groups, however for our purposes we need to sometimes create "Security Groups" aka "Mail Enabled Security Groups". These are equivalent to creating groups using the following option revealed by clicking the down arrow in Exchange Admin center:



The following command will create a security group named secgroup1, with one member user1:

```
New-DistributionGroup -Name secgroup1 -Type Security -Members user1
```

Resulting in the following:



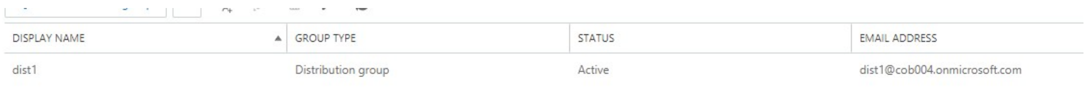
DISPLAY NAME	GROUP TYPE	STATUS	EMAIL ADDRESS
secgroup1	Security group	Active	secgroup1@cob004.onmicrosoft.com

Create a distribution list (EOL)

In a very similar manner to "Mail Enabled Security Groups", Distribution Groups or Distribution Lists are created like the following command:

```
New-DistributionGroup -Name dist1 -Members user1, user2
```

Resulting in the following:

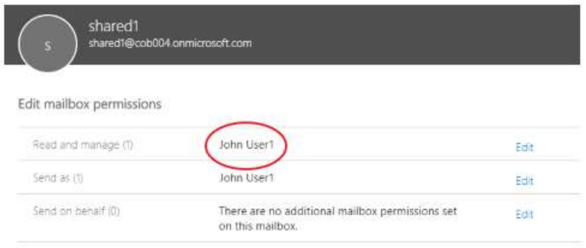


DISPLAY NAME	GROUP TYPE	STATUS	EMAIL ADDRESS
dist1	Distribution group	Active	dist1@cob004.onmicrosoft.com

Add Full Access (Read and Manage) permission for shared mailbox (EOL)

The following will add "Full Access" (aka "Read and Manage") permission for user1 to the shared1 mailbox:

```
Add-MailboxPermission shared1 -user user1 -AccessRights FullAccess
```



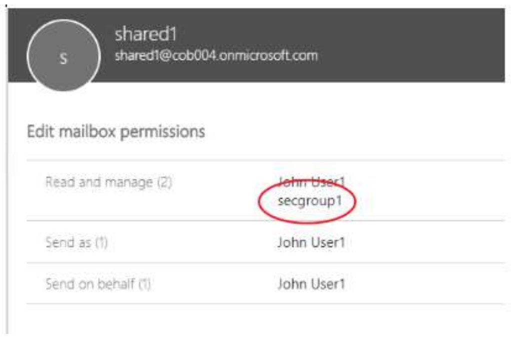
shared1
shared1@cob004.onmicrosoft.com

Edit mailbox permissions

Read and manage (1)	John User1	Edit
Send as (1)	John User1	Edit
Send on behalf (0)	There are no additional mailbox permissions set on this mailbox.	Edit

Note that user can also be a Mail Enabled Security Group, e.g.

```
Add-MailboxPermission shared1 -user secgroup1 -AccessRights FullAccess
```



shared1
shared1@cob004.onmicrosoft.com

Edit mailbox permissions

Read and manage (2)	John User1 secgroup1
Send as (1)	John User1
Send on behalf (1)	John User1

Deny permissions and removing permissions

The same command can be used with -deny to deny permissions, this is an action that can only be performed through PowerShell:

```
Add-MailboxPermission shared1 -user user1 -AccessRights FullAccess -deny
```

To see all the mailbox permissions and to remove a permission we can then run the following:

```
Get-MailboxPermission shared1
Remove-MailboxPermission shared1 -user user1 -AccessRights FullAccess -deny -Confirm:$false
```

Viewing all permissions

Get-MailboxPermission cmdlet will show the permissions on a mailbox, the following command can be used to filter out permissions we are not concerned with:

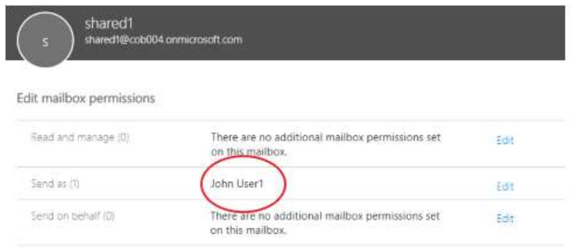
```
Get-MailboxPermission shared1 | where {$_.user.tostring() -ne "NT AUTHORITY\SELF" -and $_.IsInherited -eq $false}  | ft -autosize
```

Add Send As permission for shared mailbox (EOL)

The following will add "Send As" permission for user1 to the shared1 mailbox:

```
Add-RecipientPermission shared1 -AccessRights SendAs -Trustee user1 -Confirm:$false
```

Resulting in the following:



shared1
shared1@cob004.onmicrosoft.com

Edit mailbox permissions

Read and manage (0)	There are no additional mailbox permissions set on this mailbox.	Edit
Send as (1)	John User1	Edit
Send on behalf (0)	There are no additional mailbox permissions set on this mailbox.	Edit

Removing

The above permission can be removed with the following:

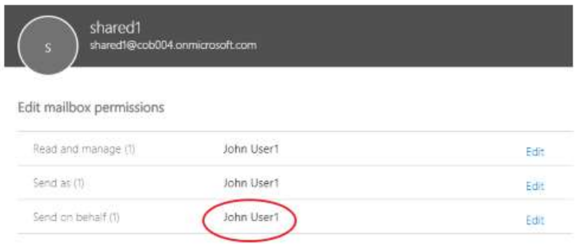
```
Remove-RecipientPermission shared1 -AccessRights SendAs -Trustee user1 -Confirm:$false
```

Add Send On Behalf permission for shared mailbox (EOL)

The following will add "Send On Behalf" permission for user1 to the shared1 mailbox:

```
Set-Mailbox shared1 -GrantSendOnBehalfTo user1
```

Resulting in the following:



shared1
shared1@cob004.onmicrosoft.com

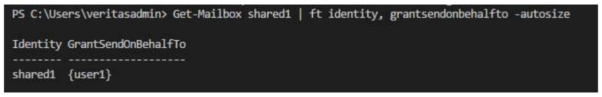
Edit mailbox permissions

Read and manage (1)	John User1	Edit
Send as (1)	John User1	Edit
Send on behalf (1)	John User1	Edit

Viewing

The Send On Behalf permission can be viewed with the following command:

```
Get-Mailbox shared1 | ft identity, grantsendonbehalfto -autosize
```



```
PS C:\Users\veritasadmin> Get-Mailbox shared1 | ft identity, grantsendonbehalfto -autosize

Identity GrantSendOnBehalfTo
-------- -------------------
shared1  {user1}
```

Removing

```
Set-Mailbox shared1 -GrantSendOnBehalfTo @{remove="user1"}
```

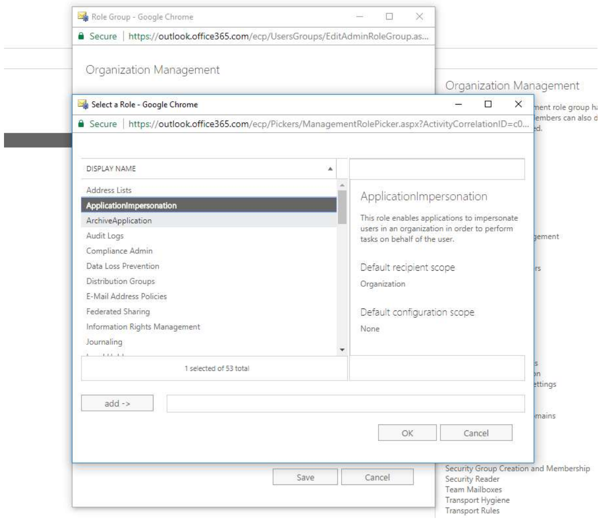
For an explanation of this multivalued property syntax see:
https://blogs.technet.microsoft.com/dstrome/2011/05/29/multivalued-properties-in-exchange-2010/

Management Role Assignment (EOL)

This command is useful in the setup of a new Office 365 account:

```
Enable-OrganizationCustomization
New-ManagementRoleAssignment -Name "ApplicationImpersonation-Organization Management" -Role "ApplicationImpersonation" -SecurityGroup "Organization Management"
```

It is the equivalent of the steps in the Admin center to add ApplicationImpersonation to Organization Management that is required during the setup of Office 365.



Role Group - Google Chrome
Secure | https://outlook.office365.com/ecp/UsersGroups/EditAdminRoleGroup.as...

Organization Management

Select a Role - Google Chrome
Secure | https://outlook.office365.com/ecp/Pickers/ManagementRolePicker.aspx?ActivityCorrelationID=c0...

DISPLAY NAME
Address Lists
ApplicationImpersonation
ArchiveApplication
Audit Logs
Compliance Admin
Data Loss Prevention
Distribution Groups
E-Mail Address Policies
Federated Sharing
Information Rights Management
Journaling

1 selected of 53 total

ApplicationImpersonation
This role enables applications to impersonate users in an organization in order to perform tasks on behalf of the user.

Default recipient scope
Organization

Default configuration scope
None