



GoateePFE

Blog of Microsoft Premier Field Engineer Ashley McGlone featuring PowerShell scripts for the real world

Home About GoateePFE

Three Steps to Migrate Group Policy Between Active Directory Domains or Forests Using PowerShell

 Ashley McGlone April 28, 2014

Rate this article

33



Three Steps Ahead

Have you ever wished that you had three legs? Imagine how much faster you could run. Today we are going to look at three steps to migrating GPOs between domains or forests with PowerShell. Now that is fast!

The Problem

Have you ever wanted to **copy** all of your production **Group Policy Objects (GPOs)** into a lab for testing? Do you have to copy GPOs between domains or forests? Do you need to migrate them to another environment due to an acquisition, merger, or divestiture? These are common problems for many administrators.

There are VBScripts provided with the **Group Policy Management Console (GPMC)**, but that is so "last decade". (*Really. They were published in 2002.*) What about WMI filters, OU

Follow Us

Sharing Links

[Follow @GoateePFE](#)

[Tweet](#)

Popular Tags

- PowerShell
- Active Directory
- AD
- Desired State Configuration
- Forensics
- DSC
- SID History
- v3
- GPO
- Group Policy
- SIDhistory
- PowerShell Saturday
- Windows Server 2012
- Groups
- Token Size
- Microsoft

links, login scripts, and embedded credentials? I've drafted a **PowerShell** module to do this with speed and style. This post discusses the pitfalls, preparations, and scripts for a successful GPO migration.

Real-World Scenario

Recently I worked with a customer who had mirrored dev, test, and prod Active Directory forests. They had the same accounts, groups, OUs, and GPOs in all three places. Then they had another version of the same dev, test, prod environment for a separate application. That is two sets of three forests, both with identical GPOs. Their current process for copying policies was manually backing up and importing the GPOs, which is how TechNet tells you to do it. At this scale, however, they were in need of an automated solution. Enter PowerShell.

Scripting Options

When automating Group Policy with the tools in the box you have three options:

1. Group Policy Management Console (GPMC) [VBScripts](#) (circa 2002)
2. [GroupPolicy PowerShell module](#) (Windows Server 2008 R2 and above, installed with GPMC)
3. [GPMgmt.GPM COM object](#) which is the secret sauce behind #1 and #2

VBScript. Yeah. That worked great all those years ago. I know. That's what I used day-in-day-out before PowerShell. But this is a new era. If you are still relying on VBScript, then it is time for an intervention from your peers.

My default choice is always to use the cmdlets out-of-the-box. And that is what I tried to do for the most part. However, while developing this solution I ran into a number of limitations with the GroupPolicy module cmdlets. I'll detail those below.

Behind the VBScripts and the cmdlets there is a COM object called "GPMgmt.GPM". Here is a list of the methods exposed by the object:

Windows 8

ADMT

Get-ADObject

Sites

Archives

[September 2017](#) (1)

[August 2017](#) (3)

[July 2017](#) (1)

[June 2017](#) (1)

[May 2017](#) (1)

[April 2017](#) (1)

[March 2017](#) (1)

[All of 2017](#) (11)

[All of 2016](#) (14)

[All of 2015](#) (15)

[All of 2014](#) (18)

[All of 2013](#) (14)

[All of 2012](#) (21)

[All of 2011](#) (20)

[All of 2010](#) (5)

Can you help me? Yes!

If you would like to have me or another Microsoft PFE visit your company and assist with the ideas presented in this blog post, then contact your Microsoft Premier Technical Account Manager (TAM) for booking information.

For more information about becoming a Microsoft Premier customer view the [information page](#) or contact us using the

[web form](#). Tell them GoateePFE sent you.

```
PS C:\> New-Object -ComObject GPMgmt.GPM | Get-Member | Select-Object Name
```

```
Name
----
CreateMigrationTable
CreatePermission
CreateSearchCriteria
CreateTrustee
GetBackupDir
GetBackupDirEx
GetClientSideExtensions
GetConstants
GetDomain
GetMigrationTable
GetRSOP
GetSitesContainer
InitializeReporting
InitializeReportingEx
```

For example, the Get-GPResultantSetOfPolicy cmdlets calls the GetRSOP method of this COM object. However, we do not have full cmdlet coverage. There are no cmdlets for working with GPO migration tables. Therefore I studied the migration table VBScripts and essentially converted them to PowerShell. The VBScripts have great value as templates for how to use this COM object. It's just not cool to rely on VBScript for much else these days.

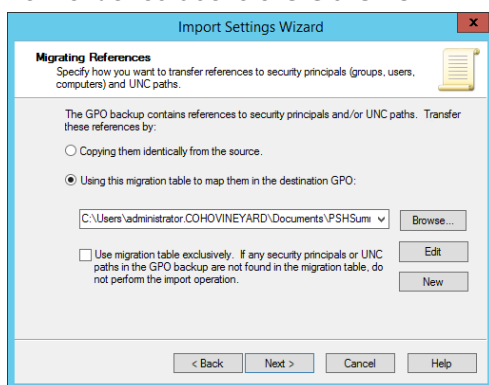
GPO Scripting Challenges

When I first sat down to tackle GPO migration I found the convenient cmdlet Copy-GPO. Game over, right? Just use the cmdlet. Oh, how I wish it were that easy. To make a very long story very short here is a summary of the challenges I encountered:

- **Copy-GPO requires both source and destination domains to be online.**

That means we cannot use it for disconnected dev, test, prod forest scenarios. No problem. I'll just use Backup-GPO and Import-GPO...

- **Backup-GPO/Import-GPO does not have the -CopyACL switch from Copy-GPO.** Now I have to find another way to migrate permissions. No problem. I'll just use the Get-GPPermission/Set-GPPermission cmdlets...
- **Set-GPPermission will not set *deny* entries, only *allow*.** Seriously? Some shops rely on deny. I had to write my own code for this piece, and it was quite involved. However, I used the opportunity to translate permissions based on the migration table, so that made it more robust in the end.
- As mentioned above **there are no**



cmdlets for Group Policy Migration

Tables. This is a necessary evil for most GPO migrations. Restricted groups, user rights assignment, script paths, etc. can be buried down in the policies. [Migration tables](#) tell the import how to translate accounts and paths in policies to the new domain. Usually creating a migration table is a manual process with an ancient GUI tool. I automated the whole thing using a simple CSV file where you can specify search/replace values to automatically update the automatically generated migration table.

- Import-GPO has a parameter to use a migration table, but it **forces the option from the GUI which requires all accounts to be in the migration table.** I left this one as-is. You can work around this by adjusting the migration table or fudging accounts.
- **Neither Copy-GPO nor Import-GPO support WMI filter migration.** After extensive research I discovered that WMI

filter scripting may require a registry hack and a DC reboot due to a “system owned object” *feature*. This one is the ugliest of them all, and I decided to leave it alone. Bin Yi from Microsoft has posted a [PowerShell module on the TechNet Script Gallery for migrating WMI filters](#). Feel free to use his code if you need this functionality. Backup-GPO puts all the WMI filter data into the backup, but writing it back to the new environment is the challenge. I’ll tackle this later if I have demand for it.

In this case the old saying is true, “It is never as easy as it looks.”

The Process

If there ever were a case for automation this is it. The export process allows us to do multiple GPOs simultaneously, and some of the import steps are optional. Even so, it is quite involved. Here is the complete, manual GPO migration process:

1. Export GPOs from source domain
2. Copy export files to destination domain
3. Create and tweak migration table
4. Manually recreate WMI filters in destination
5. Remove GPOs of same name in destination
6. Import GPOs to destination domain
7. Manually reassign WMI filters
8. Copy permissions (and sync SYSVOL permissions)
9. Link GPOs to OUs
10. Set link properties (enabled, enforced, etc.)

Now imagine repeating that effort... multiple times... by hand... without making any mistakes... without forgetting a step... and keeping your sanity.

Beginner Tip: If you have never done a GPO backup and import from the GUI, then I suggest you start there first. That will give you a better idea of the overall process. You will want to

click the option for the migration table so that you understand it as well.

The Solution

My mission is to make things simple for Microsoft customers. I was able to reduce the entire manual process down to a new PowerShell module and a CSV file. Here is an outline of the new module cmdlets involved. You will notice these correlate directly to the process steps above (except for WMI not supported in this release).

- Start-GPOExport
 - Invoke-BackupGPO
 - (Backup-GPO)
 - Export-GPPermission
- Start-GPOImport
 - New-GPOMigrationTable
 - Show-GPOMigrationTable
 - Test-GPOMigrationTable
 - Invoke-RemoveGPO
 - (Remove-GPO)
 - Invoke-ImportGPO
 - (Import-GPO)
 - Import-GPPermission
 - Import-GPLink

Let's break this down into three steps, well four if you count the setup, or maybe five if you count extra tinkering.

Step 0 – Setup

In the source domain and destination domain you want a workstation or member server with the following basic requirements:

- PowerShell version 2 or above
- Remote Server Administration Tools (RSAT)
 - Active Directory module
 - Group Policy module
 - GPMC

On your machine set up a working folder where you copy the PowerShell files from this blog post. The download link is at the bottom of the article. *By the way, you will usually need to*

unblock the file(s) after download.

I developed this on a Windows 8.1 client running PowerShell v4 and tested it on Windows Server 2008 R2 (PSv2), Windows Server 2012 (PSv3), and Windows Server 2012 R2 (PSv4).

Step 1 – Migration Table CSV File

We will call this the “migration table CSV file”. It is not a GPO migration table, but it feeds the automation process behind building and updating the migration table. Before we run the migration code we need to create a simple CSV file that maps source domain references to the destination domain. Here is an example that is included with the code:

```
Source          Destina
tion           tion
-----
----
wingtiptoy     cohovin
s.local       eyard.com    Domain
wingtiptoy     cohovin
s              eyard        Domain
\\wingtiptoy   \\cohov
s.local\      ineyard.com\ UNC
\\wingtiptoy   \\cohov
s              ineyard\     UNC
```

Notice there are short name (NetBIOS) and long name (FQDN) entries for each domain and for both “Domain” and “UNC” type. You can add other values for server names in UNC paths, etc. This is my suggested minimum. You will want one of these files for each combination of source/destination domains where you are migrating GPOs. Make copies of the sample and modify them to your needs.

Step 2 – Export

The ZIP download includes a sample calling script for the export. All you have to do is update the working folder path, modify the domain and server names, and then edit the

Where-Object line to query the GPO(s) you want to migrate.

```
Set-Location "C:\Temp\GPOMigration\"

Import-Module GroupPolicy

Import-Module ActiveDirectory

Import-Module ".\GPOMigration.psm1"
-Force

# This path must be absolute, not relative
$Path = $PWD # Current folder specified in Set-Location above

$SrcDomain = 'wingtiptoy.local'

$SrcServer = 'dca.wingtiptoy.local'

$DisplayName = Get-GPO -All -Domain $SrcDomain -Server $SrcServer |

    Where-Object {$_.DisplayName -like '*test*'} |
    Select-Object -ExpandProperty DisplayName

Start-GPOExport `
    -SrcDomain $SrcDomain `
    -SrcServer $SrcServer `
    -DisplayName $DisplayName `
    -Path $Path
```

Run the script. This calls the necessary module functions to create the GPO backup and export the permissions. Note that the permissions are listed in the GPO backup, but there is no practical way to decipher them. (Trust me. Long story.) In this case we're going to dump the permissions to a simple CSV that gets written into the same GPO backup folder.

The working folder will now include a subfolder with the GPO backup. Copy the entire working folder to your destination domain working machine.

Step 3 – Import

This is where most of the fancy foot work takes place, but I've reduced it to "one big button" if that meets your needs. The ZIP download includes a sample calling script for the import. This time you have to update the working folder path, modify the domain and server names, update the backup folder path, and then update the migration table CSV path to point to the file you created in Step 1 above.

Note: Be sure not to confuse the source and destination domain/server names. It would be unfortunate if you got those backwards when working in a production environment. Just sayin'. You've been warned.

```
Set-Location "C:\Temp\GPOMigration\"

Import-Module GroupPolicy

Import-Module ActiveDirectory

Import-Module ".\GPOMigration.psm1"
-Force

# This path must be absolute, not relative
$Path = $PWD # Current folder specified in Set-Location above

$BackupPath = "$PWD\GPO Backup wingtiptoy.local 2014-04-23-16-37-31"

$DestDomain = 'cohovineyard.com'

$DestServer = 'cvdcr2.cohovineyard.com'
$MigTableCSVPath = '.\MigTable_sample.csv'

Start-GPOImport `
  -DestDomain $DestDomain `
  -DestServer $DestServer `
  -Path $Path `
  -BackupPath $BackupPath `
  -MigTableCSVPath $MigTableCSVPath `
  -CopyACL
```

Run the script. This calls the necessary module functions to import each GPO from the backup

and put everything back in place in the destination domain. After the script finishes review the output. Check for any errors. Verify the results in the destination domain using GPMC. You can always rerun the script as many times as you like, making adjustments each time.

The working folder will now include a *.migtable file for the GPO migration table. You can view and edit this, but be aware that the default logic in Start-GPOImport will create a new one each time. Using Start-GPOImport requires to have the same accounts in the source and destination domains. You can adjust the migration table and instead use Invoke-ImportGPO directly with your custom migration table. Most likely the migration table will take some time to smooth out. You'll catch on.

Also be aware that by default Start-GPOImport removes any existing GPOs with the same name. This is by design. Remember that you can tweak the Start-GPOImport function to suit your own needs.

Step 4 – Free Style

Once you get the hang of the process I encourage you to dive into the Start-GPOImport function contained in the module. It is pre-set to do a full import. Your needs will likely vary from this template. Use the syntax from this function to build your own import routine tailored to your requirements.

Summary

In a nut shell I've taken a multiple step manual process and condensed it down to three simple steps that execute quickly in PowerShell. I agree that it is a pain to update paths in the calling script and copy files around. On the bright side it is still way faster than the manual alternative.

As always when you are copying scripts from the internet make sure that you understand what the script will do before you run it. Test it in a lab before using it in production. Open up the GPOMigration.psm1 module file and skim through the code. Review the full help content

for each function. You will learn more PowerShell and get ideas for your own scripts.

I'd love to hear how this script module has helped you. Please use the comments below to ask questions and offer feedback. Put your best foot forward with PowerShell!

Get the script [here on the TechNet Script Center](#).

Read the [follow up post](#) with WMI filter migration supported.

Tags [GPO](#) [Group Policy](#) [PowerShell](#)

Comments (33)

Name *

Email *

Website



anonymouscommenter

May 27, 2014 at 8:08 pm

I know this post is a little late, but I wanted to offer some helpful information that I picked up at the PowerShell Summit last month. This post is packed with links to keep you surfing high-value PowerShell content for days.

[Reply](#)



anonymouscommenter

May 28, 2014 at 12:17 pm

Pingback from AD: Three Steps to Migrate Group Policy Between Active Directory Domains or Forests Using PowerShell | MS Tech BLOG

[Reply](#)

anonymouscommenter



August 11, 2014 at 12:05 pm

This week I am presenting a session on GPO migration at TechMentor Redmond 2014. This is an expanded version of the session I gave at the PowerShell Summit back in April. I received feedback in April that WMI filters must be supported before this would

[Reply](#)



Edward van Biljon

August 15, 2014 at 6:50 am

thank you

[Reply](#)



anonymouscommenter

October 15, 2014 at 1:00 pm

This is absolutely great. I noticed that if I use Group Policy Reference to add domain users to local groups the domain name remains same that it was on source domain. Workaround is to use old style policies (Windows SettingsSecurity SettingsRestricted Groups)

[Reply](#)



anonymouscommenter

October 30, 2014 at 4:33 pm

Hi There

Scripts almost working for us but we have one query..... what syntax should we use in Migration Table CSV file to allow us to map AD Group Names that are different in source and destination domains?

Do we use "Domain" or "UNC" type or something else?

[Reply](#)



anonymouscommenter

November 5, 2014 at 3:02 pm

Excellent Solution – Finding it very useful. Many Thanks

[Reply](#)



anonymouscommenter

November 5, 2014 at 4:12 pm

Thanks, it's very good solution.

[Reply](#)



anonymouscommenter

November 6, 2014 at 8:41 am

Good Document.....Thanks..

[Reply](#)



anonymouscommenter

January 6, 2015 at 9:01 am

Thanks. Would be great if there is a way to automate the migration csv creation

[Reply](#)



anonymouscommenter

January 8, 2015 at 9:13 pm

It's people like you, Ashley, that make my job so much easier. Thanks a lot for this.

[Reply](#)



anonymouscommenter

February 11, 2015 at 7:40 pm

Almost perfect but there's something a little screwy going on for me... When restoring GPOs that have "Authenticated users: Read" it is being restored in destination domain with "Authenticated users: Read, Apply group policy". This means GPOs with security filtering applied to one user/group are getting applied to everyone until I manually fix the permissions. Is no-one else experiencing similar?

[Reply](#)



anonymouscommenter

March 19, 2015 at 11:31 am

I am coping Richard H as his problem is 100 % exactly the same as mine.

Almost perfect but there's something a little screwy going on for me... When restoring GPOs that have "Authenticated users: Read" it is being restored in destination domain with "Authenticated users: Read, Apply group policy". This means GPOs with security filtering applied to one user/group are getting applied to everyone until I manually fix the permissions. Is no-one else experiencing similar?

[Reply](#)



Lishron

March 19, 2015 at 11:48 am

It seems to add the authenticated users to every GPO I move.

[Reply](#)



Anonymous

June 20, 2016 at 10:46 am

The exporting script works great, however the import script does not link the GPO's in the correct order. Has anyone else seen this?

[Reply](#)

anonymouscommenter

April 28, 2015 at 11:16 am



I got around the authenticated users issue by going through all the ones after I import and removing them.

Import-Module ActiveDirectory

Set-Location -Path 'AD:'

Set-GPPermission -Name 'Policy I Just Imported' -PermissionLevel None -
TargetName 'Authenticated Users' -TargetType Group

[Reply](#)



anonymouscommenter

[May 8, 2015 at 4:40 pm](#)

This is working out great for me as I am bringing in 65 GPOs from 4 different domains. Really helps on the time factor. Thanks for sharing!

[Reply](#)



anonymouscommenter

[June 16, 2015 at 1:14 pm](#)

This works as charm for me, Thanks for posting this!

[Reply](#)



anonymouscommenter

[September 14, 2015 at 6:43 pm](#)

Awesome solution!! thanks!!

[Reply](#)



anonymouscommenter

[October 9, 2015 at 11:35 am](#)

This is gold, I had such a struggle with the classic scripts and the old migration table.

One point here though, for me it didn't work with domain local groups (it did with global and universal). Anyhow this should be an easy fix.

Thanks a lot !

[Reply](#)



anonymouscommenter

[November 25, 2015 at 5:15 pm](#)

Hi. I am using your scripts for a relatively small interforest migration. I ran into a problem with the GPPermissions export functionality. In my case I am running the scripts from a DC in the target domain for both the export and import. I think others

may not run into this particular issue. The CSV file is generated without any content and 0KB size. Here's a sample error:

Get-ADObject : Cannot find an object with identity:
'cn={86FF4668-CBFB-4630-848C-

```
193D0BCE52BA},cn=policies,cn=system,DC=sourcedomain,DC=corp' under:
'DC=ad,DC=targetdomain,DC=com'.
At C:\TempGPOMigrationGPOMigrationGPOMigration.psm1:129 char:17
+ $ACL = (Get-ADObject -Identity $GPO.Path -Properties NTSecurityDescripto
...
+
~~~~~
~~~~~
+ CategoryInfo          : ObjectNotFound: (cn={86FF4668-CB...
                        =surgem,DC=corp:ADObject) [Get-ADObject], ADIdentityN
                        otFoundException
+ FullyQualifiedErrorId :
ActiveDirectoryCmdlet:Microsoft.ActiveDirectory.Management.ADIdentityNotFo
undException,M
icrosoft.ActiveDirectory.Management.Commands.GetADObject
```

I looked at the Get-ADObject command and saw that it has the -Server option. So, I set up a quick test script as follows:

```
$GPO = Get-GPO -Server dc01.surgem.corp -Domain surgem.corp -Name
ParticularGPO

(Get-ADObject -Server dc01.sourcedomain.corp -Identity $GPO.Path -
Properties NTSecurityDescriptor |
Select-Object -ExpandProperty NTSecurityDescriptor).Access
```

That command worked fine, so I just added "-Server \$SrceServer" to the Get-ADObject command at line 129 in GPOMigration.psm1. Subsequently, the Call-GPOExport.ps1 script ran without error and I now have a populated CSV file.

[Reply](#)



anonymouscommenter
[November 25, 2015 at 5:18 pm](#)

Oops. In my last comment, I attempted to mask the domain names by changing the name to sourcedomain, but I missed a couple in my haste.

[Reply](#)



marks62
[January 3, 2016 at 8:43 pm](#)

I'm getting these errors:

```
Get-GPO : The RPC server is unavailable. (Exception from HRESULT:
0x800706BA)
At C:\Usersmark\DesktopGPOMigrationCall-GPOExport.ps1:71 char:16
+ $DisplayName = Get-GPO -All -Domain $SrceDomain -Server $SrceServer |
+
~~~~~
~~~
+ CategoryInfo          : NotSpecified: (:) [Get-GPO], COMException
```

```
+ FullyQualifiedErrorId :  
System.Runtime.InteropServices.COMException,Microsoft.GroupPolic  
  
Start-GPOExport : Cannot bind argument to parameter 'DisplayName' because  
it is null.  
At C:\Users\mark\Desktop\GPOMigrationCall-GPOExport.ps1:78 char:18  
+ -DisplayName $DisplayName`  
+ ~~~~~  
+ CategoryInfo          : InvalidData: (:) [Start-GPOExport],  
ParameterBindingValidationEx  
+ FullyQualifiedErrorId :  
ParameterArgumentValidationErrorNullNotAllowed,Start-GPOExport  
  
RPC service is running. This is being run in Server 2012 Standard.  
  
$DisplayName = Get-GPO -All -Domain $SrcDomain -Server $SrcServer |  
Where-Object {$_.DisplayName -like '*test*'} |  
Select-Object -ExpandProperty DisplayName  
  
I'm unsure what the Where-Object "*test*" should be. Any help would be  
appreciated.  
Mark
```

[Reply](#)



Daniel

October 3, 2016 at 5:09 pm

did you figured it out? test is like an alias to search for in your
gpo's

[Reply](#)



Daniel

October 3, 2016 at 5:10 pm

better call it an wildcard

[Reply](#)



Michael J. Mahoney

January 27, 2016 at 5:12 pm

Before I started working at my new company, they had setup their QA and Dev domains using the SAME domain FQDN on with both development domains on disparate networks. Granted the UNC pathnames are unique, but can this import script target the right Destination with just the UNC unique?

[Reply](#)



Alex Chung

June 30, 2016 at 2:09 am

Just want to say thank you for writing this up. I was almost at the point of setting up each GPO again manually, so this has saved me a lot of time!!

[Reply](#)



Daniel

October 3, 2016 at 5:08 pm

Hey there, thanks a lot for your work, i really appreciate this!!!

As a german user I encountered the problem that i recieve a lot of errors while importing:

WARNUNG: Error. Cannot set: 'Dom?nen-Admins' on 'Windows-V3.0-C-DC'

WARNUNG: Error. Cannot set: 'DOM?NENCONTROLLER DER ORGANISATION' on 'Windows-V3.0-C-DC'

and so on. Anybody has an idea?

[Reply](#)



Adam Bravo

April 13, 2017 at 12:15 pm

How does this compare with

<https://blogs.technet.microsoft.com/heyscriptingguy/2014/01/05/use-powershell-to-import-group-policy-objects/> ?

[Reply](#)



Jens Schroeder

September 19, 2017 at 4:22 am

Hi

A few suggestions for improvements.

The script handles authenticated users in a way that make this account always added. In addition, the script fails at the rights assignment.

Therefore, I suggest adding -and \$ MigID.Type -ne "Unknown" in function Import-GPPPermission, so it will be like

```
If ($ MigID -and $ MigID.Type -ne "Unknown") {
```

```
# Find the AD object based on the type listed in the MigTable
```

This will direct authenticated users etc. outside the migtable handling, which will fail in looking up the SID.

Also, I suggest, these two lines must be added immediately after import-gpo in function Invoke-ImportGPO, to prevent authenticated users from being automaticly assigned:

```
Set-GPPPermissions -Domain $ DestDomain -Server $ DestServer -Name $  
GPMBackup.GPODisplayName -PermissionLevel 'GPOread' -TargetName  
'Authenticated Users' -TargetType Group -Replace -Confirm: $ false
```

```
Set-GPPPermissions -Domain $ DestDomain -Server $ DestServer -Name $  
GPMBackup.GPODisplayName -PermissionLevel 'GPOread' -TargetName  
'Domain Computers' -TargetType Group -Replace -Confirm: $ false
```

The second line can be omitted.

[Reply](#)



sherif osama

November 2, 2017 at 2:43 am

I had the below problems

From domain vfdev.com removing GPO: Test1

The “Test1” GPO was not found in the vfdev.com domain.

Parameter name: gpoDisplayName

The output error

he object of type

“Microsoft.PowerShell.Commands.Internal.Format.FormatStartData” is not valid or not in the correct sequence. This is likely caused by a user-specified “format-list” command which is conflicting with the default formatting.

+ CategoryInfo : InvalidData: (:) [out-lineoutput], InvalidOperationException

+ FullyQualifiedErrorId :

ConsoleLineOutputOutOfSequencePacket,Microsoft.PowerShell.Commands.OutLineOutputCommand

[Reply](#)



Angel L. Cruz

February 20, 2018 at 4:00 pm

THIS SIMPLY WORKS!

A community solution that simply works without me fixing anything! Thanks!

Thanks! Thanks!

To all of you out there, simply FOLLOW INSTRUCTIONS!

[Reply](#)



LIHU

May 1, 2018 at 5:28 pm

If you’re running export from an out-of-domain server, here’s a post v1.1 code correction :

GPOMigration.psm1#129 :

\$ACL = (Get-ADObject -Identity \$GPO.Path -Properties NTSecurityDescriptor | becomes

\$ACL = (Get-ADObject -Identity \$GPO.Path -Properties NTSecurityDescriptor - Server \$SrceServer |

[Reply](#)

