

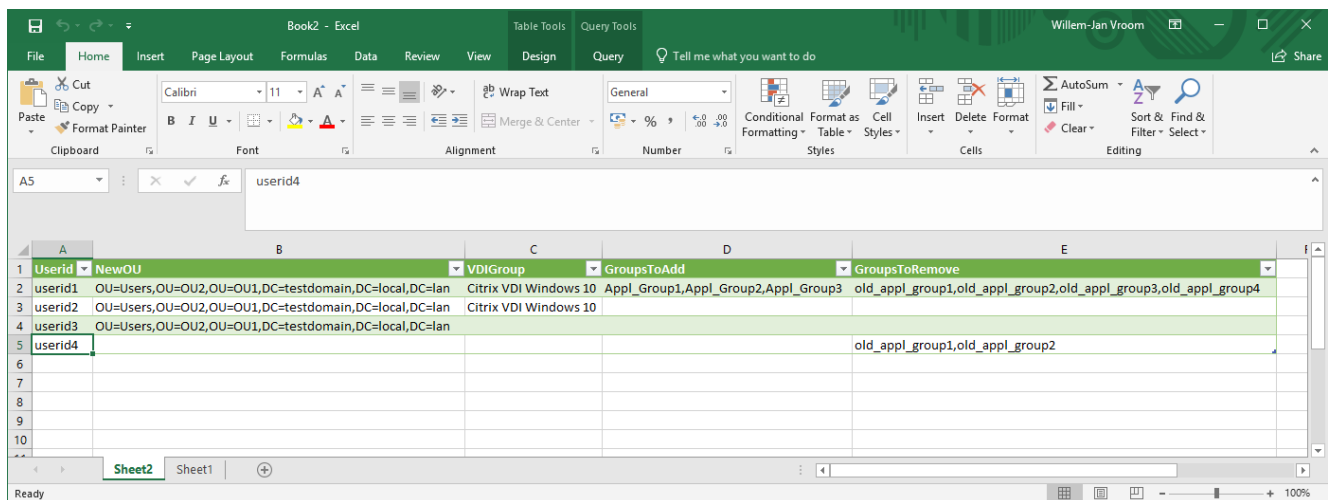
Migrate users from Windows 7 to Windows 10 automatically

For a migration from Windows 7 to Windows 10 it is needed to do the following with the users' account:

- Move it to a different OU
- Add it to a new VDI group
- Add it to the new application groups
- Remove it from the previous application groups
- Clear the profile path, if needed.

To do this, I created a Powershell script.

The users who need to be migrated from Windows 7 to Windows 10 are in a csv file:



Userid	NewOU	VDIGroup	GroupsToAdd	GroupsToRemove
user1	OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan	Citrix VDI Windows 10	Appl_Group1,Appl_Group2,Appl_Group3	old_appl_group1,old_appl_group2,old_appl_group3,old_appl_group4
user2	OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan	Citrix VDI Windows 10		
user3	OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan			
user4				old_appl_group1,old_appl_group2

- The column **Userid** contains the userid
- The column **NewOU** contains the OU where the user should be moved to. This column can be empty.
- The column **VDIGroup** contains the new VDI group. This column can be empty.
- The column **GroupsToAdd** contains the groups the user should be added to. Split the groups with a comma. This column can be empty.
- The column **GroupsToRemove** contains the groups the user

should be removed from. Split the groups with a comma.
This column can be empty.

An example:

```
"Userid","NewOU","VDIGroup","GroupsToAdd","GroupsToRemove"
"userid1","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
"Citrix VDI Windows 10",
"Appl_Group1,Appl_Group2,Appl_Group3",
"old_appl_group1,old_appl_group2,old_appl_group3,old_appl_group4"
"userid2","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
"Citrix VDI Windows 10","",""
"userid3","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
","",""
"userid4","","","old_appl_group1,old_appl_group2"
```

The are some parameters:

- **FileWithUseridsInCSVFormat:** CSV Filename that contains all the userids that should be migrated. If not mentioned than the script name is used.
- **ProductionRun:** Use the swith ProductionRun to modify. If not specified, the script is run in test mode.
- **FullCleanUp:** Use the swith FullCleanUp to remove all unneeded groups.
- **ClearProfilePath:** Use the swith ClearProfilePath to clear the profile path.

The script:

```
<#
```

```
.NOTES
```

```
=====
=====
=====
```

```
Created with:      Windows PowerShell ISE
Created on:        06-September-2018
Created by:        Willem-Jan Vroom
Organization:
```

Filename: Migrate users (v01).ps1

=====

=====

=====

.DESCRIPTION:

This script prepares the users for the migration from Windows 7 to Windows 10.

.USAGE:

Create a CSV file with the following layout:

```
"userid","NewOU","VDIGroup","GroupsToAdd","GroupsToRemove"
"userid1","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
"Citrix VDI Windows 10",
"Appl_Group1,Appl_Group2,Appl_Group3",
"old_appl_group1,old_appl_group2,old_appl_group3,old_appl_group4"
"userid2","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
"Citrix VDI Windows 10","",""
"userid3","OU=Users,OU=OU2,OU=OU1,DC=testdomain,DC=local,DC=lan",
"","",""
"userid4","","","","old_appl_group1,old_appl_group2"
```

Explanation:

Userid1: moved to the OU testdomain.local.lan\OU1\OU2\Users, and added to the groups Citrix VDI Windows 10, Appl_Group1, Appl_Group2 and Appl_Group3 and also removed from the

groups old_appl_group1, old_appl_group2, old_appl_group3 and old_appl_group4.

Userid2: moved to the OU testdomain.local.lan\OU1\OU2\Users, and added to the group Citrix VDI Windows 10.

Userid3: only moved to the OU testdomain.local.lan\OU1\OU2\Users.

Userid4: the user is removed from the groups old_appl_group1 and old_appl_group2.

.PARAMETERS:

-FileWithUseridsInCSVFormat:

The filename with the users to be modified.

-ProductionRun

If the switch is used then ActiveDirectory is modified. If not used then a test run is done, and Active Directory is not modified.

-FullCleanUp

If this switch is used then the following groups will be removed from the users' account:

- * All gg_appl groups
- * WM-Users

-ClearProfilePath

If this switch is used then the profile path will be cleared.

.VERSION HISTORY:

v0.1:

- * Initial version.

#>

param

```
(
[Parameter(HelpMessage="CSV Filename that contains all the
userids that should be migrated. If not mentioned than the
script name is used.")]
[String] $FileWithUseridsInCSVFormat="",

[Parameter(HelpMessage="Use the swith ProductionRun to modify.
If not specified, the script is run in test mode.")]
[Switch] $ProductionRun,

[Parameter(HelpMessage="Use the swith FullCleanUp to remove
all unneeded groups.")]
[Switch] $FullCleanUp,

[Parameter(HelpMessage="Use the swith ClearProfilePath to
clear the profile path.")]
[Switch] $ClearProfilePath
```

)

#

=====
=====
=====

Function block

#

=====
=====
=====

Function Write-EntryToResultsFile

{

<#

.NOTES

=====
=====
=====

Created with: Windows PowerShell ISE

Created on: 03-August-2018

Created by: Willem-Jan Vroom

Organization:

Functionname: Write-EntryToResultsFile

=====
=====
=====

.DESCRIPTION:

This function adds the success or failure information to the array that contains the log information.

#>

param

(

\$strUserId,

\$ErrorMessage = "",

\$Action = ""

)

```
$Record = [ordered] @{"Username" = "";"Action"=
"";"Testmode"="";"Error"= ""}
$Record."Username" = $strUserid
$Record."Action" = $Action
$Record."Testmode" = -not($ProductionRun)
$Record."Error" = $ErrorMessage
$objRecord = New-Object PSObject -Property $Record
$Global:arrTable += $objRecord
}
```

```
Function Remove-ProfilePathFromUserProfileInAD
{
```

```
<#
```

```
.NOTES
```

```
=====
=====
=====
```

```
Created with: Windows PowerShell ISE
```

```
Created on: 06-September-2018
```

```
Created by: Willem-Jan Vroom
```

```
Organization:
```

```
Functionname: Remove-ProfilePathFromUserProfileInAD
```

```
=====
=====
=====
```

```
.DESCRIPTION:
```

This function clears the ProfilePath from AD.

```
#>
```

```
param
(
    $strUserid
)
Try
{
    $strUserDN = (Get-ADUser -Identity
$strUserid).distinguishedName
    Set-ADUser -Identity $strUserDN -Clear profilePath -
```

```
WhatIf:(-not($ProductionRun)) -Confirm:$false
    Write-EntryToResultsFile -strUserid $strUserid -Action
"Clear profile path"
    }
    Catch
    {
        Write-EntryToResultsFile -strUserid $strUserid -
ErrorMessage $_.Exception.Message -Action "Clear profile path"
        Continue
    }
}
```

```
Function Move-ADUserToOtherOU
```

```
{
```

```
<#
```

```
.NOTES
```

```
=====
=====
=====
```

```
Created with:      Windows PowerShell ISE
Created on:        06-September-2018
Created by:        Willem-Jan Vroom
Organization:
Functionname:      Move-ADUserToOtherOU
```

```
=====
=====
=====
```

```
.DESCRIPTION:
```

```
This function moves the user to another OU.
```

```
#>
```

```
param
```

```
(
    $strUserid,
    $strDestinationOU
)
```

```
Try
```

```
{
```

```

    if($strDestinationOU.Length -gt 0)
    {
        $strUserDN = (Get-ADUser -Identity
$strUserid).distinguishedName
        Move-ADObject -Identity $strUserDN -TargetPath
$strDestinationOU -WhatIf:(-not($ProductionRun)) -
Confirm:$false
        Write-EntryToResultsFile -strUserid $strUserid -Action
"Move AD User to OU $strDestinationOU."
    }
}
Catch
{
    Write-EntryToResultsFile -strUserid $strUserid -
ErrorMessage $_.Exception.Message -Action "Move AD User to OU
$strDestinationOU."
    Continue
}
}

```

```

Function Add-ADMemberToGroup
{

```

```
<#
```

```
.NOTES
```

```

=====
=====
=====

```

```

Created with:    Windows PowerShell ISE
Created on:      06-September-2018
Created by:      Willem-Jan Vroom
Organization:
Functionname:    Add-ADMemberToGroup

```

```

=====
=====
=====

```

```
.DESCRIPTION:
```

```
This function adds a user to an AD group.
```

```
#>
```



```

param
(
    $strUserid,
    $strADGroupName
)
Try
{
    if($strADGroupName.Length -gt 0)
    {
        $strUserDN = (Get-ADUser -Identity
$strUserid).distinguishedName
        Add-ADGroupMember -Identity $strADGroupName -Members
$strUserDN -WhatIf:(-not($ProductionRun)) -Confirm:$false
        Write-EntryToResultsFile -strUserid $strUserid -Action
"Add AD group $strADGroupName to user."
    }
}
Catch
{
    Write-EntryToResultsFile -strUserid $strUserid -
ErrorMessage $_.Exception.Message -Action "Add AD group
$strADGroupName to user."
    Continue
}
}

```

```

Function Remove-ADMemberFromGroup
{

```

```
<#
```

```
.NOTES
```

```

=====
=====
=====
Created with:      Windows PowerShell ISE
Created on:        06-September-2018
Created by:        Willem-Jan Vroom
Organization:
Functionname:      Remove-ADMemberFromGroup
=====
=====

```

=====

.DESCRIPTION:

This function removes a user from an AD group.

#>

```
param
(
    $strUserid,
    $strADGroupName
)
Try
{
    if($strADGroupName.Length -gt 0)
    {
        $strUserDN = (Get-ADUser -Identity
$strUserid).distinguishedName
        Remove-ADGroupMember -Identity $strADGroupName -Members
$strUserDN -WhatIf:(-not($ProductionRun)) -Confirm:$false
        Write-EntryToResultsFile -strUserid $strUserid -Action
"Remove AD group $strADGroupName from user."
    }
}
Catch
{
    Write-EntryToResultsFile -strUserid $strUserid -
ErrorMessage $_.Exception.Message -Action "Remove AD group
$strADGroupName from user."
    Continue
}
}
```

#

=====

=====

End function block

#

=====

```

=====

#
=====
=====
=====

# Declares the variables.
#
=====
=====
=====

    $valCounter                = 1
    $Global:arrTable           = @()
    $strActivity                = "Modifying users in
Active Directory."
    $arrGroupMustContainForFullCleanUp = @("gg_appl","WM-Users")
    $strCurrentPath            = Split-Path -parent
$MyInvocation.MyCommand.Definition
    $strCurrentFile            =
$MyInvocation.MyCommand.Name

#
=====
=====
=====

# Define the CSV Import File.
#
=====
=====
=====

if($FileWithUseridsInCSVFormat.Length -eq 0)
{
    $strCSVFileName = $strCurrentFile -Replace ".ps1",".csv"
}
else
{
    if($FileWithUseridsInCSVFormat.ToLower().IndexOf(".csv") -
eq -1)

```

```
{
    $FileWithUseridsInCSVFormat = $FileWithUseridsInCSVFormat
+ ".csv"
}
$strCSVFileName = $FileWithUseridsInCSVFormat
}
```

```
#
=====
=====
=====
```

```
# Check if the string $strCSVFileName is a path. In that case,
nothing has to be done.
# In case it is not a path, then the current location should
be added.
```

```
#
=====
=====
=====
```

```
if (-not(Split-Path($strCSVFileName)))
{
    $strCSVFileName = $strCurrentPath + "\" + $strCSVFileName
}
```

```
#
=====
=====
=====
```

```
# Define the log file. This log file contains all the results.
#
=====
=====
=====
```

```
$strLastPartOfFileName = "_" + (Get-Date).ToString('F') +
".csv"
$strLastPartOfFileName = $strLastPartOfFileName -replace
":","-"
$strCSVLogFileSucces = $strCSVFileName -Replace ".csv",
$strLastPartOfFileName
```

```

If(Test-Path $strCSVLogFileSucces)
{
Remove-Item $strCSVLogFileSucces
}

#
=====
=====
=====
# Read the CSV file.
#
=====
=====
=====
if(Test-Path $strCSVFileName)
{
$arrUserids = @(Import-Csv $strCSVFileName)
}
Else
{
Write-Host "The import file $strCSVFileName does not
exists."
Exit 1
}
#
=====
=====
=====
# Process the users in the CSV file.
#
=====
=====
=====

Clear-Host

If(-not($ProductionRun))
{
$strActivity = $strActivity + " (RUNNING IN TEST MODE)"
}

```

```

ForEach($objUser in $arrUserids)
{
    $strUserid = $objUser.Userid
    Write-Progress -Activity $strActivity -Status "Processing
user $strUserid" -PercentComplete ($valCounter /
$arrUserids.Count * 100)
    Try
    {
        $strNewOU          = $objUser.NewOU
        $strVDIGroup      = $objUser.VDIGroup
        $arrGroupsToAdd   = $objUser.GroupsToAdd.Split(",")
        $arrGroupsToRemove = $objUser.GroupsToRemove.Split(",")
    }
    Catch
    {
        Write-Host "There is something wrong with the CSV filename
$strCSVFileName while processing $strUserid."
        Exit 1
    }
    # Clear the profile path
    If($ClearProfilePath)
    {
        Remove-ProfilePathFromUserProfileInAD -strUserid
$strUserid
    }
    # Move the user to another OU:
    Move-ADUserToOtherOU -strUserid $strUserid -
strDestinationOU $strNewOU
    # Add the user to the new VDI group:
    Add-ADMemberToGroup -strUserid $strUserid -strADGroupName
$strVDIGroup

    # Add the user to various groups:
    ForEach($objGroupToAdd in $arrGroupsToAdd)
    {
        Add-ADMemberToGroup -strUserid $strUserid -strADGroupName
$objGroupToAdd
    }
    # Remove the user to various groups:
    ForEach($objGroupToRemove in $arrGroupsToRemove)
    {

```

```

        Remove-ADMemberFromGroup -strUserid $strUserid -
strADGroupName $objGroupToRemove
    }
    # Full Clean Up
    if($FullCleanUp)
    {
        $arrGroups = @(Get-ADUser $strUserid -Properties
MemberOf).MemberOf
        foreach($objGroup in $arrGroups)
        {
            $strGroup=(Get-ADGroup $objGroup).Name
            foreach($objGroupMustContain in
$arrGroupMustContainForFullCleanUp)
            {
                $strGroupMustContain =
$objGroupMustContain.ToString().ToLower()
                if($strGroup.ToLower().IndexOf($strGroupMustContain) -eq
0)
                {
                    Remove-ADMemberFromGroup -strUserid $strUserid -
strADGroupName $strGroup
                }
            }
        }
    }

    $valCounter++
    Sleep 2
}

Sleep 2

#
=====
=====
=====
# Write the results to the csv file.
#
=====
=====
=====

```

```
If($Global:arrTable.Count -gt 0)
{
    $Global:arrTable | Export-Csv $strCSVLogFileSucces -
NoTypeInfoInformation
}
Else
{
    Write-Host "Something went wrong while writing the logfile
$strCSVLogFileSucces. Maybe nothing to report..."
}
```

Any feedback to improve this script is appreciated. You can download the scripts here:

1. [Migrate users \(v01\)](#)