**Import Users from a CSV File into Active Directory using PowerShell**

[Dennis Suhanovs](http://www.flexecom.com/author/dennis-suhanovs/) June 29, 2013 [Active Directory](http://www.flexecom.com/category/active-directory/), [PowerShell](http://www.flexecom.com/category/powershell/)

Manipulating Active Directory objects programmatically isn’t anything new or exciting, however, I had to assist a client recently with a request where PowerShell was the preferred delivery method (normally I prefer to write managed code in one of the .NET languages).

**Objectives**

1. Process a large input file (in CSV format)
2. Edit existing user object with attributes from the file
3. Assign manager attribute from the file

**Approach**

Importing a CSV file is dead simple in PowerShell:

Import-CSV $importfile

Where $importfile is a variable set to the path of the script. We create a loop that goes through each imported line by piping the import into ForEach-Object command:

Import-CSV $importfile | ForEach-Object {

#do something here

}

To copy fields from the CSV file into variables, we access the properties of the imported object using “this object” reference, $\_ – like so:

$sam = $\_.Users

$manager = $\_.Manager

$location = $\_.Location

$company = $\_.Company

$department = $\_.Department

Where “Users”, “Manager”, “Location”, etc are names of the CSV columns contained in the first row of the CSV file.

Next, while in the ForEach-Object loop, we call Active Directory and set user properties using the following command: Set-ADUser.

Set-ADUser -Identity $sam department=$department

This command assumes that the user already exists in the directory, if your use case is different, for example you are not sure if the user exists, you could test the presence of the user object via Try/Catch block:

Try { $exists = Get-ADUser -LDAPFilter "(sAMAccountName=$sam)" }

Catch { }

If ($exists)

{

Set-ADUser -Identity $sam department=$department

}

So the next issue, is that not all attributes of AD objects can be modified using this direct approach, like shown above with department attribute. Only commonly used attributes are accessible this way, as per [Microsoft Set-ADUser documentation](http://technet.microsoft.com/en-us/library/ee617215.aspx). Other attributes will need to be accessed using -Replace parameter. When supplying values to certain parameters, you have to test for nulls and ensure that new values you pass to -Replace are not nothing.

If ($location) { Set-ADUser -Identity $sam -Replace @{l=$location} }

If ($company) { Set-ADUser -Identity $sam -Replace @{company=$company} }

Finally, if you are working with special attributes that are stored as distinguishedName references to other objects ([linked value attributes](http://msdn.microsoft.com/en-us/library/windows/desktop/ms677270%28v=vs.85%29.aspx)), you may have to call out to the directory to obtain the dn of the object being referred to. This is true of Manager, Reports, and MemberOf attributes. Here we need to also test that the object we are about to reference actually exists.

Try { $exists = Get-ADUser -LDAPFilter "(sAMAccountName=$manager)" }

Catch { }

If ($exists)

{

Get-ADUser -Identity $sam | Set-ADUser -Manager $\_.Manager

}

**Import and Modify Existing Objects Script**

Now let’s put the things above into a PowerShell script that imports a CSV file and updates attributes of user objects, that already exist in the directory.

Import-Module ActiveDirectory

$path = Split-Path -parent $MyInvocation.MyCommand.Definition

# ENSURE THAT IMPORT FILE IS IN THE SAME DIRECTORY AS THIS SCRIPT

#

$importfile = $path + "\sample.csv"

$logfile = $path + "\logfile.txt"

#

# DO NOT EDIT AFTER THIS LINE

Function ImportUsers

{

Import-CSV $importfile | ForEach-Object {

$sam = $\_.Users.ToLower()

$manager = $\_.Manager

$output = "User: " + $sam

Try {

$location = $\_.Location

$company = $\_.Company

$department = $\_.Department

If ($location) { Set-ADUser -Identity $sam -Replace @{l=$location} }

If ($department) { Set-ADUser -Identity $sam department=$department }

If ($company) { Set-ADUser -Identity $sam -Replace @{company=$company} }

If ($manager) {

Try { $exists = Get-ADUser -LDAPFilter "(sAMAccountName=$manager)" }

Catch { }

If ($exists) {

Get-ADUser -Identity $sam | Set-ADUser -Manager $\_.Manager

}

}

$output + " updated successfully" | Out-File $logfile -append

}

Catch {

$output + " update failed: " + $error[0].ToString() | Out-File $logfile -append

}

}

}

ImportUsers

**Dump User Attributes Script**

In some cases, it may be useful to write out the attributes you are about to mass-update. This could be handy in HR matters, or to have a paper record for change management purposes. Suppose that we have the same input CSV file as the one used in the import script above. In this script below, we import the contents of the input file, but only read the Users field. We then query Active Directory for presence of each user, and if the user is located, we write out the attributes into a separate text file.

Import-Module ActiveDirectory

$path = Split-Path -parent $MyInvocation.MyCommand.Definition

# ENSURE THAT IMPORT FILE IS IN THE SAME DIRECTORY AS THIS SCRIPT

#

$importfile = $path + "\sample.csv"

$logfile = $path + "\logfile.txt"

$before = $path + "\before.txt"

#

# DO NOT EDIT AFTER THIS LINE

Function ImportUsers

{

"users,company,department,location,manager" | Out-File $before -append

Import-CSV $importfile | ForEach-Object {

$sam = $\_.Users

Try {

$u = Get-ADuser $sam -Properties samaccountname,department,city,company,manager

$u.samaccountname + "," + $u.company + "," + $u.department + "," + $u.city + "," + $u.manager | Out-File $before -append

}

Catch {

"Error: failed to process user " + $sam + " - " + $error[0].ToString() | Out-File $logfile -append

}

}

}

ImportUsers

Note how -Properties parameter is used in Get-ADUser commandlet to specify which attributes of the user object we want to load.

**Import and Create New Active Directory Accounts Script**

Suppose that we are still working with the same input file, but instead of editing existing user objects we need to create new ones.

Instead of Set-ADUser commandlet we will be using New-ADUser commandlet. Another twist in the script below is setting of a new temporary password. User objects can’t (or rather, should not) have blank passwords. Finally, when we create a new account, by default it will be created as disabled account; we override this by passing $True boolean value with -Enabled parameter.

$initialpassword = "some complex password"

$setpass = ConvertTo-SecureString -AsPlainText $initialpassword -force

New-ADUser $sam -DisplayName $sam -Company $\_.Company -Department $\_.Department -City $\_.Location -AccountPassword $setpass -Enabled:$True

In addition to setting of the account status and password, we may want to move the account to some target organizational unit (OU).

$location = "OU=Test,DC=corp,DC=domain,DC=com"

$dn = (Get-ADUser $sam).DistinguishedName

Move-ADObject -Identity $dn -TargetPath $location

That’s all there is to it. Now the full script:

Import-Module ActiveDirectory

$path = Split-Path -parent $MyInvocation.MyCommand.Definition

$count = 0

# ENSURE THAT IMPORT FILE IS IN THE SAME DIRECTORY AS THIS SCRIPT

#

$importfile = $path + "\sample.csv"

$location = "OU=Test,DC=corp,DC=domain,DC=com"

$initialpassword = "some complex password"

$logfile = $path + "\logfile.txt"

#

# DO NOT EDIT AFTER THIS LINE

Function ImportUsers

{

Import-CSV $importfile | ForEach-Object {

$sam = $\_.Users.ToLower()

$manager = $\_.Manager.ToLower()

$output = "User: " + $sam

Try {

$setpass = ConvertTo-SecureString -AsPlainText $initialpassword -force

New-ADUser $sam -DisplayName $sam -Company $\_.Company -Department $\_.Department -City $\_.Location -AccountPassword $setpass -Enabled:$True

$dn = (Get-ADUser $sam).DistinguishedName

Move-ADObject -Identity $dn -TargetPath $location

Try { $exists = Get-ADUser -LDAPFilter "(sAMAccountName=$manager)" }

Catch { }

If ($exists)

{

Get-AdUser -Identity $sam | Set-AdUser -Manager $\_.Manager

}

$output + " imported successfully" | Out-File $logfile -append

}

Catch {

$output + " failed to import: " + $error[0] | Out-File $logfile -append

}

}

}

ImportUsers

Bottom of Form

**Tags**

[Active Directory Assessment](http://www.flexecom.com/tag/active-directory-assessment/) [Active Directory health check](http://www.flexecom.com/tag/active-directory-health-check/) [Active Directory programming](http://www.flexecom.com/tag/active-directory-programming/) [ADBPA](http://www.flexecom.com/tag/adbpa/) [ADFS](http://www.flexecom.com/tag/adfs/) [ADST](http://www.flexecom.com/tag/adst/) [ADTD](http://www.flexecom.com/tag/adtd/) [Azure](http://www.flexecom.com/tag/azure/) [Best Practices](http://www.flexecom.com/tag/best-practices/) [DirSync](http://www.flexecom.com/tag/dirsync/) [Disaster Recovery](http://www.flexecom.com/tag/disaster-recovery/) [DL360 Gen8](http://www.flexecom.com/tag/dl360-gen8/) [Forefront Identity Manager](http://www.flexecom.com/tag/forefront-identity-manager/) [GPMC](http://www.flexecom.com/tag/gpmc/) [Group Policy](http://www.flexecom.com/tag/group-policy/) [hardware](http://www.flexecom.com/tag/hardware/) [Hypervisor](http://www.flexecom.com/tag/hypervisor/) [InstallUtil.exe](http://www.flexecom.com/tag/installutil-exe/) [MBSA](http://www.flexecom.com/tag/mbsa/) [netsh](http://www.flexecom.com/tag/netsh/) [Office 365](http://www.flexecom.com/tag/office-365/) [Querying AD](http://www.flexecom.com/tag/querying-ad/) [Querying DNS](http://www.flexecom.com/tag/querying-dns/) [Remote Access](http://www.flexecom.com/tag/remote-access/) [SCOM agents](http://www.flexecom.com/tag/scom-agents/) [SCOM alerts](http://www.flexecom.com/tag/scom-alerts/) [SCOM connector](http://www.flexecom.com/tag/scom-connector/) [SCOM queries](http://www.flexecom.com/tag/scom-queries/) [SCOM SDK](http://www.flexecom.com/tag/scom-sdk/) [SCOM subscription](http://www.flexecom.com/tag/scom-subscription/) [SCSM query](http://www.flexecom.com/tag/scsm-query/) [SCSM SDK](http://www.flexecom.com/tag/scsm-sdk/) [Security](http://www.flexecom.com/tag/security/) [self-hosted web service](http://www.flexecom.com/tag/self-hosted-web-service/) [SharePoint migration](http://www.flexecom.com/tag/sharepoint-migration/) [SQL](http://www.flexecom.com/tag/sql/) [SSO](http://www.flexecom.com/tag/sso/) [Virtualization](http://www.flexecom.com/tag/virtualization/) [Visual Studio Express 2012](http://www.flexecom.com/tag/visual-studio-express-2012/) [WCF](http://www.flexecom.com/tag/wcf/) [WIDS](http://www.flexecom.com/tag/wids/) [Windows API](http://www.flexecom.com/tag/windows-api/) [Windows Server 2012](http://www.flexecom.com/tag/windows-server-2012/) [WMI](http://www.flexecom.com/tag/wmi/) [WSUS](http://www.flexecom.com/tag/wsus/)

**Categories**

* [Active Directory](http://www.flexecom.com/category/active-directory/)
* [Active Directory Certificate Services](http://www.flexecom.com/category/active-directory-certificate-services/)
* [Hyper-V](http://www.flexecom.com/category/hyper-v/)
* [Office 365](http://www.flexecom.com/category/office-365/)
* [PowerShell](http://www.flexecom.com/category/powershell/)
* [SCOM 2012](http://www.flexecom.com/category/scom/)
* [SCSM 2012](http://www.flexecom.com/category/scsm-2012/)
* [Sharepoint / WSS](http://www.flexecom.com/category/sharepoint-wss/)
* [Uncategorized](http://www.flexecom.com/category/uncategorized/)
* [Web Services](http://www.flexecom.com/category/web-services/)
* [Windows Management Instrumentation](http://www.flexecom.com/category/windows-management-instrumentation/)
* [Windows Server 2012](http://www.flexecom.com/category/windows-server-2012/)
* [WSUS 2012](http://www.flexecom.com/category/wsus-2012/)

**Recent Posts**

* [Deploying Enterprise PKI on Windows Server 2012 R2](http://www.flexecom.com/deploying-enterprise-pki-on-windows-server-2012-r2/)
* [GAL Segmentation in Office 365](http://www.flexecom.com/gal-segmentation-in-office-365/)
* [How to Obtain Tier Queue Name Programmatically in SCSM 2012](http://www.flexecom.com/how-to-obtain-tier-queue-name-programmatically-in-scsm-2012/)
* [Deploy Dirsync on Windows Server 2012 for Office 365 (Part 4)](http://www.flexecom.com/deploy-dirsync-on-windows-server-2012-for-office-365-part-4/)
* [Install ADFS 2.1 on Windows Server 2012 for Office 365 (Part 3)](http://www.flexecom.com/install-adfs-2-1-on-windows-server-2012-for-office-365-sso-part-3/)
* [Install ADFS 2.1 on Windows Server 2012 for Office 365 (Part 2)](http://www.flexecom.com/install-adfs-2-1-for-office-365-on-windows-server-2012-part-2/)
* [Install ADFS 2.1 on Windows Server 2012 for Office 365 (Part 1)](http://www.flexecom.com/install-adfs-2-1-on-windows-server-2012-for-office-365-part-1/)
* [Office 365 Single Sign-On with ADFS 2.1 on Windows Server 2012](http://www.flexecom.com/office-365-single-sign-on-with-adfs-2-1-on-windows-server-2012/)
* [Import Users from a CSV File into Active Directory using PowerShell](http://www.flexecom.com/import-users-from-a-csv-file-into-active-directory-using-powershell/)
* [Unable to Change Password Logging into an Azure-hosted Virtual Server](http://www.flexecom.com/unable-to-change-password-logging-into-an-azure-hosted-virtual-server/)

**Archives**

* [December 2013](http://www.flexecom.com/2013/12/) (1)
* [July 2013](http://www.flexecom.com/2013/07/) (6)
* [June 2013](http://www.flexecom.com/2013/06/) (2)
* [March 2013](http://www.flexecom.com/2013/03/) (1)
* [February 2013](http://www.flexecom.com/2013/02/) (2)
* [December 2012](http://www.flexecom.com/2012/12/) (2)
* [November 2012](http://www.flexecom.com/2012/11/) (6)
* [October 2012](http://www.flexecom.com/2012/10/) (5)
* [September 2012](http://www.flexecom.com/2012/09/) (10)

**Friends**

* [CMS Consulting Inc (alma mater)](http://www.cms.ca)
* [Dennis Suhanovs](http://www.suhanovs.com)
* [RUN CORP (consulting partner)](http://www.run-corp.com)
* [SecTor IT Security Conference](http://www.sector.ca)
* [Toronto Area Security Klatch](http://www.task.to)

**Inspirations**

* [Rework](http://37signals.com/rework)
* [TED: Ideas Worth Spreading](http://www.ted.com)

**Notable Projects**

* [3.75 kW solar panel setup & blog](http://www.greentoronto.me/a-few-more-pictures-of-my-solar-array/)
* [Exchange 2000 Design book](http://www.amazon.com/MCSE-Exchange-2000-Design-Exam/dp/1588800261/ref=sr_1_3?s=books&ie=UTF8&qid=1347233818&sr=1-3&keywords=suhanovs)
* [Infrastructure Guardian](http://www.infrastructureguardian.com)
* [Windows Server 2003 Active Directory book](http://www.amazon.com/Windows-Server-Active-Directory-Infrastructure/dp/0072223197/ref=la_B001K8EHDW_1_2?ie=UTF8&qid=1347233750&sr=1-2)
* [Windows Server 2008 Active Directory book](http://http:/www.amazon.com/Windows-Server-Active-Directory-Services/dp/0071599274/ref=la_B001K8EHDW_1_1?ie=UTF8&qid=1347233629&sr=1-1)