

# Managing Active Directory Users with Windows PowerShell

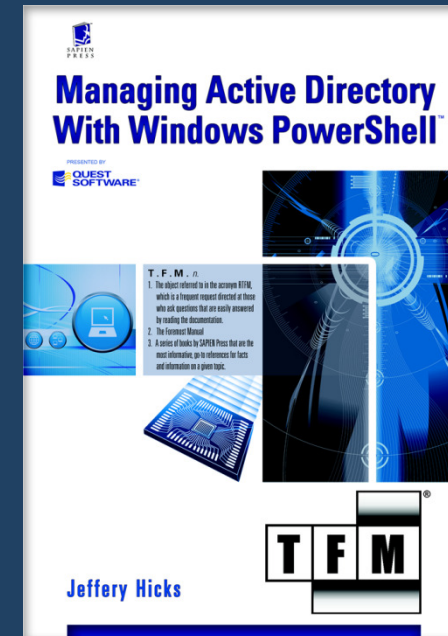
**Jeffery Hicks**  
**SAPIEN Technologies, Inc.**

Pre-requisites for this presentation:

- 1) Active Directory Experience
- 2) PowerShell Experience

Level: Intermediate

- Creating Users
- Adding to Groups
- Managing Passwords
- Modifying Users
- Managing the Masses



# Creating Users



## Pick a Technique

- Active Directory Services Interface (ADSI)
- PowerShell Community Extensions
- Quest Active Directory Cmdlets

# PowerShell & ADSI

- PowerShell uses .NET Framework
- [System.DirectoryServices.DirectoryEntry]
- Use the [ADSI] type accelerator

[ADSI]\$OU="LDAP://OU=Staff,DC=MyCo,DC=local"



- Connect to parent container
- Create("user","CN=User name)
- Set SAMAccountname
- Set password

# ADSI Options

- Set User Principal Name
- Enable Account
- Set personal properties like first name, last name, title

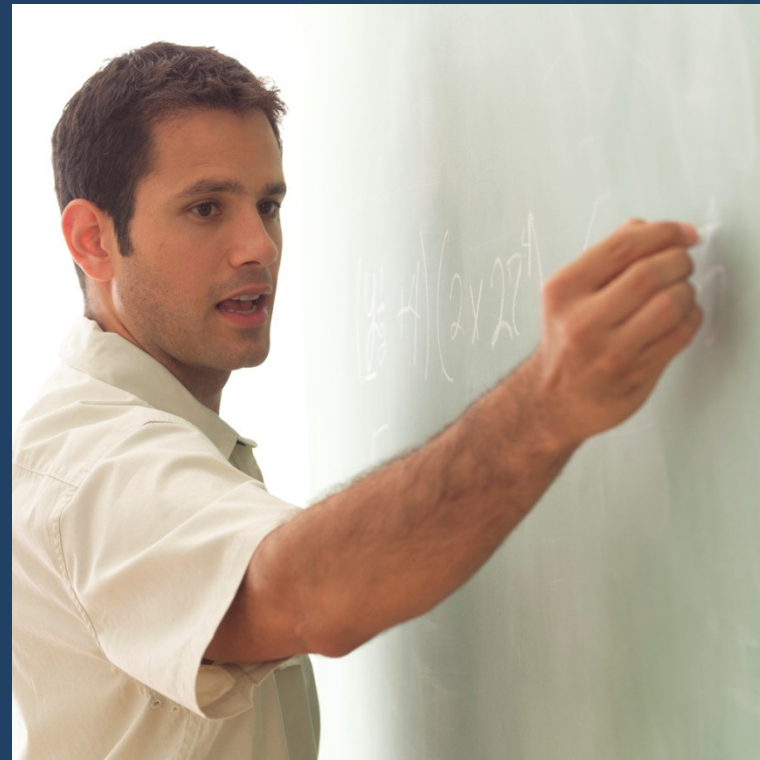
- ADSI creates a local cached copy of the account
- Use SetInfo() to commit changes to Active Directory



## Simple Solution

```
[ADSI]$OU="LDAP://OU=Staff,DC=Spacely,DC=Local"
$user=$OU.Create("user","CN=George Jetson")
$user.Put("sAMAccountName","gjetson")
$user.SetInfo()
$user.Put("UserPrincipalName",`
    "gjetson@spacely.com")
$user.SetPassword("$pr0ck3T")
$user.SetInfo()
```

- New-User.ps1



# Quest AD Cmdlets

- Download Quest PowerShell Active Directory Cmdlets
- Installs as a PSSnapin
- Help \*QAD\* to see all the cmdlets

- Creates a new user account with as many properties as you want
- Help New-QADUser –full
- Help New-QADUser -examples

## New-QADUser Sample

```
$OU="OU=Public  
    Affairs,OU=Employees,DC=Bigcompany,DC=local"  
New-QADUser -name "Michael Bloomberg" `  
    -ParentContainer $OU -samAccountName `"  
    mbloomberg " -UserPassword "P@ssw0rd" `"  
-firstname "Michael" -LastName "Bloomberg" `"  
-userprincipalname "mrb@bigcompany.com" `"  
-City "New York City" `"  
-department "Public Affairs" -Title "Liason" `"  
-company "Big Company"
```

- Demo-QADUser.ps1
- Enable-QADUser



# Adding to Groups



- Create ADSI object for group
- Add user's ADSI path
- No need for SetInfo()

## Example

```
[ADSI]$group="LDAP://CN=IT  
Admins,OU=Groups,DC=MyCo,DC=local"  
$group.Add("LDAP://CN=Jeff  
Hicks,OU=IT,DC=MyCo,DC=local")
```

- The MemberOf property for a user object will return a list of immediate group memberships

```
PS C:\> [ADSI]$User="LDAP://CN=Jeff  
Hicks,OU=Staff,DC=MyCo,DC=Local"
```

```
PS C:\> $user.MemberOf
```

# Add-QADGroupMember

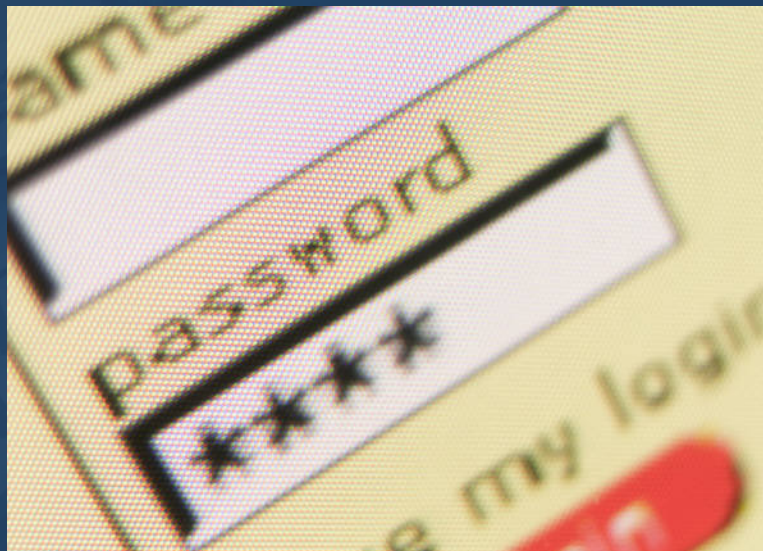
- Need group object, but you can specify it by DN, CN or SAM
- Need user object, but you can specify it by DN,CN or SAM

```
PS C:\> Add-QADGroupMember "IT Admins"  
"Jeff Hicks"
```

- Demo-QADGroupMember.ps1



# Managing Passwords



## Using ADSI

- Get the ADSI object
- Call the SetPassword() method

```
PS C:\> [ADSI]$user="LDAP://CN=Jeff  
Hicks,OU=Staff,DC=MyCo,DC=local"
```

```
PS C:\> $user.SetPassword("$ecr3+")
```

# Using Set-QADUser

- Set-QADUser *username* –UserPassword

```
PS C:\> set-qaduser "Roy Biv" -userpassword  
"P@ssw0rd"
```

# Forcing Password Change

- With ADSI, set pwdLastSet to 0

```
PS C:\> [ADSI]$user="LDAP://CN=Tom  
Foolery,OU=Staff,DC=MyCo,DC=local"
```

```
PS C:\> $User.Put("pwdLastSet",0)
```

```
PS C:\> $User.SetInfo()
```

## Using Set-QADUser

- Set-QADUser *username*  
-UserMustChangePassword

```
PS C:\> set-qaduser "Roy Biv"  
-UserMustChangePassword $true
```

```
PS C:\> set-qaduser "Roy Biv"  
-UserMustChangePassword $true  
-userpassword "P@ssw0rd"
```



# Password Properties

- Properties like non-expiring password are bit flags stored in the UserAccountControl property
- Perform a binary comparison to determine what has been set



# Flag Constants

- ADS\_UF\_SCRIPT 0x0001
- ADS\_UF\_ACCOUNTDISABLE 0x0002
- ADS\_UF\_HOMEDIR\_REQUIRED 0x0008
- ADS\_UF\_LOCKOUT 0x0010
- ADS\_UF\_PASSWD\_NOTREQD 0x0020
- ADS\_UF\_PASSWD\_CANT\_CHANGE 0x0040
- ADS\_UF\_ENCRYPTED\_TEXT\_PASSWORD\_ALLOWED 0x0080  
ADS\_UF\_DONT\_EXPIRE\_PASSWD 0x10000
- ADS\_UF\_SMARTCARD\_REQUIRED 0x40000
- ADS\_UF\_PASSWD\_EXPIRED 0x800000

# Binary Comparison

```
PS C:\> New-Variable ADS_UF_DONT_EXPIRE_PASSWD  
0x10000 -Option Constant
```

```
PS C:\> [ADSI]$user="LDAP://CN=Jack  
Frost,OU=Testing,DC=jdhitsolutions,DC=local"
```

```
PS C:\> $user.useraccountcontrol
```

```
66080
```

```
PS C:\> if ($user.useraccountcontrol[0] -band  
$ADS_UF_DONT_EXPIRE_PASSWD) {write-host  
"don't expire"}
```

```
don't expire
```

# Using Get-QADUser

```
PS C:\> if ((get-qaduser "jack  
frost").useraccountcontrol -band  
$ADS_UF_DONT_EXPIRE_PASSWD)  
{ "don't expire"  
don't expire
```

- Get-PasswordProperty.ps1
- Set-PWDNeverExpire.ps1
- Get-DomainPasswordReport.ps1

# Modifying Users



# Using ADSI

- Connect to user object
- Assign a new value to property
  - Object.Property = value
  - Object.Put(propertyname,value)
- Use SetInfo() to commit changes



## Example

```
PS C:\> $user.properties.title="Engineer"  
PS C:\> $user.put("Department","Mfg")  
PS C:\> $user.setinfo()
```

# Get-Member and ADSI

- Piping an ADSI object to get-member only shows currently defined properties
- You need to know the property name ahead of time.

- Get the user by Samaccount, DN, name or canonical name
- Many common properties are referenced via named nparameters
- No need to call SetInfo()

```
PS C:\> set-qaduser "Jack Frost"  
-Office "THX" -telephone "1138"  
-company "JDHIT"  
-title "President"  
-department "Management"
```

# Managing the Masses



# Pipelined Objects

- PowerShell is all about the objects
- Get objects
- Send them to one or more cmdlets
- Get results



# A Simple Pipeline Example



# Updating Users

- Much easier using the Quest cmdlets
- Find users objects that meet some criteria using Get-QADUser
- Pipe objects to Set-QADUser

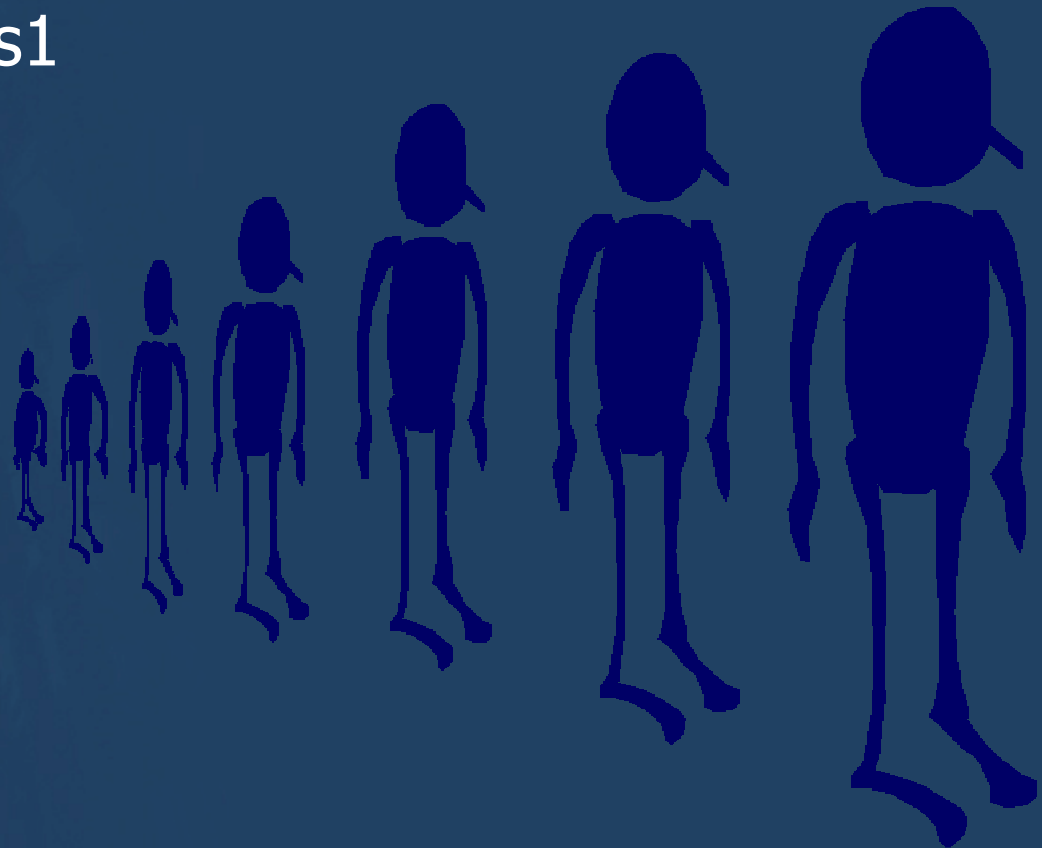
# Updating Users

```
PS C:\> get-qaduser -department "sales"  
| set-qaduser -department "Sales &  
Marketing" -city "Miami"
```

- Imports CSV formatted data into Objects
- Header line defines the objects properties
- Import-CSV *filename.csv*
- Save results to a variable or pipe immediately to other cmdlets, scripts or functions

# ADSI Provisioning

- Import-Users.ps1



# Quest Provisioning

- Import-NewUsers.ps1





- [Blog.SAPIEN.com](http://Blog.SAPIEN.com)
- [www.ScriptingAnswers.com](http://www.ScriptingAnswers.com)
- [www.PowerShellCommunity.org](http://www.PowerShellCommunity.org)
- [www.codeplex.com/PowerShellCX](http://www.codeplex.com/PowerShellCX)
- [www.quest.com/powershell/activeroles-server.aspx](http://www.quest.com/powershell/activeroles-server.aspx)
- [www.PowerGUI.org](http://www.PowerGUI.org)

- *Managing Active Directory with Windows PowerShell: TFM* (Hicks)
- *Windows PowerShell v1.0: TFM 2<sup>nd</sup> edition* (Jones & Hicks)
- *Windows PowerShell Cookbook* (Holmes)
- *Windows PowerShell in Action* (Payette)



**It's QUESTION TIME !!**

**Thank you**

- [jhicks@sapien.com](mailto:jhicks@sapien.com)
- Follow me: [twitter.com/JeffHicks](https://twitter.com/JeffHicks)

