Assignment 4:  
AD, Group Policy & System Maintenance

*Kyle Walker*

*NETW2500*

*10/28/2024*

Table of Contents

[Task 1: Create a new OU and use a PowerShell script to create users 2](#_Toc181110924)

[TranscriptA4.txt: 2](#_Toc181110925)

[NETWUserList.csv: 4](#_Toc181110926)

[NETWaddusers\_KAW.ps1: 4](#_Toc181110927)

[NETWgroupadd.ps1: 7](#_Toc181110928)

[Task 2: Managing User Environments 9](#_Toc181110929)

[Part 1: 9](#_Toc181110930)

[Part 2: 10](#_Toc181110931)

[Task 3: 10](#_Toc181110932)

[NETWcleantemp.ps1: 11](#_Toc181110933)

[Documentation and Backups: 13](#_Toc181110934)

# Task 1: Create a new OU and use a PowerShell script to create users

## TranscriptA4.txt:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Windows PowerShell transcript start

Start time: 20241028085610

Username: kaw\SysOp.KAW

RunAs User: kaw\SysOp.KAW

Configuration Name: Microsoft.PowerShell

Machine: DC01A1C20 (Microsoft Windows NT 10.0.20348.0)

Host Application: C:\Windows\system32\wsmprovhost.exe -Embedding

Process ID: 4556

PSVersion: 5.1.20348.2760

PSEdition: Desktop

PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.20348.2760

BuildVersion: 10.0.20348.2760

CLRVersion: 4.0.30319.42000

WSManStackVersion: 3.0

PSRemotingProtocolVersion: 2.3

SerializationVersion: 1.1.0.1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Transcript started, output file is C:/Reports/TranscriptA4.txt

PS C:\Users\SysOp.KAW\Documents> CommandInvocation(New-ADOrganizationalUnit): "New-ADOrganizationalUnit"

>> ParameterBinding(New-ADOrganizationalUnit): name="Name"; value="Staff"

>> ParameterBinding(New-ADOrganizationalUnit): name="Path"; value="DC=kaw,DC=netw2500,DC=ca"

PS C:\Users\SysOp.KAW\Documents> CommandInvocation(Get-ADOrganizationalUnit): "Get-ADOrganizationalUnit"

>> ParameterBinding(Get-ADOrganizationalUnit): name="Identity"; value="OU=Staff,DC=kaw,DC=netw2500,DC=ca"

City :

Country :

DistinguishedName : OU=Staff,DC=kaw,DC=netw2500,DC=ca

LinkedGroupPolicyObjects : {}

ManagedBy :

Name : Staff

ObjectClass : organizationalUnit

ObjectGUID : 115974d3-f601-4dcc-99d2-d93a1747a138

PostalCode :

State :

StreetAddress :

PS C:\Users\SysOp.KAW\Documents> CommandInvocation(Stop-Transcript): "Stop-Transcript"

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Windows PowerShell transcript end

End time: 20241028090251

## NETWUserList.csv:

FirstName,MiddleInitial,LastName,FirstInitial,Department,Office,Title

Charlotte,H,Anichka,C,Exec,Executive Offices,VP

Clare,L,Cade,C,Exec,Executive Offices,VP

Kyle,A,Walker,K,Exec,Executive Offices,CIO

Myra,Y,Dante,M,General,General Cubical Space,Admin

Jaymz,A,Dinara,J,General,General Cubical Space,Admin

Brock,R,Finn,B,General,General Cubical Space,Admin

Troy,R,Macey,T,General,General Cubical Space,Admin

Zia,I,Oleg,Z,General,General Cubical Space,Admin

Francis,R,Uta,F,General,General Cubical Space,Admin

Carlos,A,Yen,C,Sales,Sales Offices,Rep

Bruce,R,Orca,B,Sales,Sales Offices,Rep

Nina,I,Yuli,N,Sales,Sales Offices,Rep

## NETWaddusers\_KAW.ps1:

A blue screen with white text

Description automatically generated



*Share details*

# Author: Kyle Walker - W0263439

# Created: 10/28/2024

# Script created to create new users through C:/Scripts/NETWUserList.csv

# Make sure the script runs unrestricted for this PS Session

Set-ExecutionPolicy -Scope Process -ExecutionPolicy Unrestricted

# Import Active Directory module so New-ADUser and Set-ADUser will work

Import-Module ActiveDirectory

# Path to the CSV file

$csvPath = "\\MS01A1C22\Scripts\NETWUserList.csv"

# Import the CSV file

$userList = Import-Csv -Path $csvPath

# Begin a loop for each user in the csv file

foreach ($user in $userList) {

# Create the Common Name

$commonName = "$($user.FirstName) $($user.MiddleInitial) $($user.LastName)"

# Set the SAM Account Name (First Initial + Last Name)

$samAccountName = "$($user.FirstInitial)$($user.LastName)"

# Set the Display Name (Last Name, First Name)

$displayName = "$($user.LastName), $($user.FirstName)"

# Set user properties in an array to be pushed through AD

# Initialize array

$userParams = @{

# FirstName, MiddleInitial, LastName for user

Name = $commonName

# Account name per naming convention (KWalker)

SamAccountName = $samAccountName

# Principal name for our domain

UserPrincipalName = "$samAccountName@kaw.netw2500.ca"

# User's first name

GivenName = $user.FirstName

# User's last name

Surname = $user.LastName

# Set Display name as per instructions (LastName, <space>First Name) aka $displayName created above

DisplayName = $displayName

# Set users department from csv

Department = $user.Department

# Set users office from csv

Office = $user.Office

# Set users title from csv

Title = $user.Title

# Creating first password

AccountPassword = (ConvertTo-SecureString "Netw2500@2024" -AsPlainText -Force)

# Enabling the account, ensuring it is able to be accessed and logged into

Enabled = $true

# The path to our Staff OU where we will be creating the users

Path = "OU=Staff,DC=kaw,DC=netw2500,DC=ca"

}

# Create the user in Active Directory using the array created above

New-ADUser @userParams

# Set the user to change password at next login

Set-ADUser -Identity $samAccountName -ChangePasswordAtLogon $true

}



*Successful Run*

## NETWgroupadd.ps1:

# Author: Kyle Walker - W0263439

# Created: 10/28/2024

# This script creates three groups (General\_group, Sales\_group, Exec\_group)

# and adds users to the appropriate groups based on department membership.

# Set the execution policy to unrestricted for the current session only

Set-ExecutionPolicy -Scope Process -ExecutionPolicy Unrestricted

# Import the Active Directory module for AD commands like New-ADGroup and Add-ADGroupMember

Import-Module ActiveDirectory

# Define an array of group names to create

$groups = @("General\_group", "Sales\_group", "Exec\_group")

# Path to the CSV file containing user information

$csvPath = "\\MS01A1C22\Scripts\NETWUserList.csv"

# Import the CSV file with user information

$userList = Import-Csv -Path $csvPath

# Loop through each group name in the $groups array

foreach ($group in $groups) {

# Check if the group already exists to prevent duplicates

if (-not (Get-ADGroup -Filter {Name -eq $group} -ErrorAction SilentlyContinue)) {

# Create the group if it does not exist

New-ADGroup -Name $group -GroupScope Global -Path "OU=Groups,DC=kaw,DC=netw2500,DC=ca"

Write-Output "Group $group created successfully."

} else {

Write-Output "Group $group already exists. Skipping creation."

}

}

# Loop through each user in the CSV file

foreach ($user in $userList) {

# Get the SAM Account Name from CSV data

$samAccountName = "$($user.FirstInitial)$($user.LastName)"

# Determine the group based on the user's department

switch ($user.Department) {

"General" { $targetGroup = "General\_group" }

"Sales" { $targetGroup = "Sales\_group" }

"Exec" { $targetGroup = "Exec\_group" }

default { $targetGroup = $null } # No group assigned if department is not matched

}

# If a target group is assigned based on the department

if ($targetGroup) {

# Add the user to the appropriate group

# Including a try function for troubleshooting

try {

Add-ADGroupMember -Identity $targetGroup -Members $samAccountName

Write-Output "User $samAccountName added to $targetGroup."

} catch {

Write-Output "Error adding user $samAccountName to $targetGroup"

}

} else {

Write-Output "No target group for user $samAccountName. Skipping."

}

}

# Task 2: Managing User Environments

## Part 1:

A screenshot of a computer

Description automatically generated

*Successful Password Change for Cyen*

A screenshot of a computer screen

Description automatically generated

*Unsuccessful Password Change for BOrca*

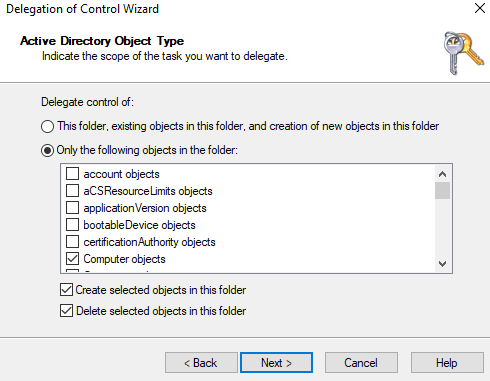
## Part 2:

A screenshot of a computer

Description automatically generated

Select all users in Staff, Open Properties, Select Profile tab, check Home Folder, Connect H: Drive and input the Shared path to the $Home share on Secondary Domain Controller, click Apply.

# Task 3:



A screenshot of a computer screen

Description automatically generated

*Computer Delegation for Domain Join* (may be unnecessary)

## NETWcleantemp.ps1:

# Created by: Kyle Walker - W0263439

# Creation date: 10/29/2024

# For the purposes of gathering and deleting temporary files

# And leaving a text file as evidence and record of their deletion

# Set temp file locations

$tempfiles = @("C:\Users\\*\AppData\Local\Temp\\*", "$env:windir\Temp\\*")

# Force removal of all files and subfolders, ignore errors

# Start a loop

foreach ($temp in $tempfiles) {

# Find each item in tempfiles recursively, if an error occurs, ignore (silently)

Remove-Item -Path $temp -Recurse -Force -ErrorAction SilentlyContinue

}

# Create a cleanlog.txt file in each temp location with the current date and time

# Create variable for date and time for simplicity

$currentDateTime = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

# Create variable to be written to cleanlog report

$logContent = "Cleanup performed on: $currentDateTime"

# Create same variable as tempfiles, but different to avoid overlap with first loop

$locations = @("C:\Users\\*\AppData\Local\Temp\", "$env:windir\Temp\")

# Loop through $locations/$tempfiles

foreach ($location in $locations) {

# Ensure cleanlog.txt is created in same directory as temp files

$logPath = Join-Path -Path $location -ChildPath "cleanlog.txt"

# Use string variable $logContent to populate cleanlog.txt and create the file

Set-Content -Path $logPath -Value $logContent -Force

}

**Script saved to** [**\\dc01a1c20\SYSVOL\kaw.netw2500.ca\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Scripts\Startup**](file:///\\dc01a1c20\SYSVOL\kaw.netw2500.ca\Policies\%7b31B2F340-016D-11D2-945F-00C04FB984F9%7d\MACHINE\Scripts\Startup)

# Documentation and Backups:

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

*Domain Controller GoldCopies (RODC1 is SD01)*

A screenshot of a computer

Description automatically generated *(DC02 is actually MS01)*

A diagram of a time flow

Description automatically generated

A diagram of a time

Description automatically generatedA diagram of a time flow

Description automatically generated

*VMWare Snapshots for DC01, SD01 and MS01*