AUTOMATED AD USER CREATION USING POWERSHELL AND CSV

Streamlining user onboarding in Windows Server environments

Summary

This project demonstrates how to automate the creation of multiple Active Directory users using a PowerShell script and a CSV file. The solution simplifies IT onboarding by generating user accounts, placing them in the correct Organizational Units (OUs), and assigning them to relevant Security Groups, all without manual setup.

Tebogo Matseding

Intro

Set up Network

1st Login

Change the Server IP Address

Change the Server IP Address 2

Verify IP Address is changed

Renaming Server Name

Verify Name Has Changed

Install Active Directory Domain Services (AD DS) role

Promote the server to a Domain Controller (Get-Command -Module addsdeployment)

Promote the server to a Domain Controller (Install-ADDSForestInstallation)

Verify Installations

Import-Module ActiveDirectory

Creating Organizational Unit

Verify that OUs have been created

Creating Security Groups

Verify that SGs have been created

Create .csv file with users

Create script that automates user creation & assign them OUs & SGs

Execute Script

Verify users have been created

Filtering users

Login into Office-PC01

Rename Office-PC01

Changing IP Address and DNS

Joining Office-PC01 to domain

Office-PC01 joined the domain

Login in with user

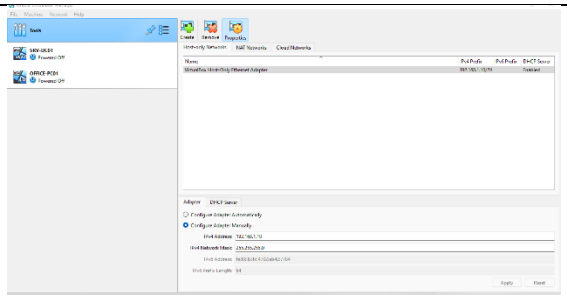
START

Intro

In this project, we set up a Windows Server 2022 environment from scratch and automate the process of creating Active Directory (AD) users using a PowerShell script and a CSV file. This simulates real-world IT workflows, especially useful in organizations onboarding many users at once.

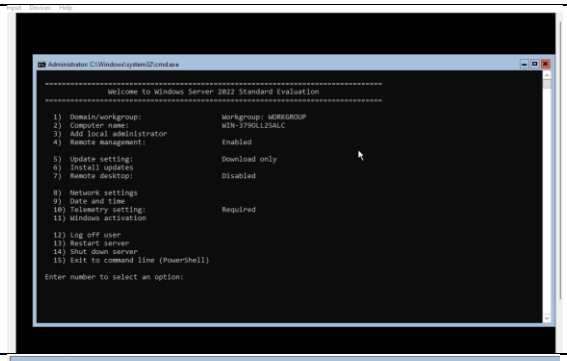
Set up Network

We start by ensuring our server is connected to a reliable and isolated network, either through a virtual switch (in VirtualBox or Hyper-V) or a real network environment. A working network connection is crucial for domain communications.



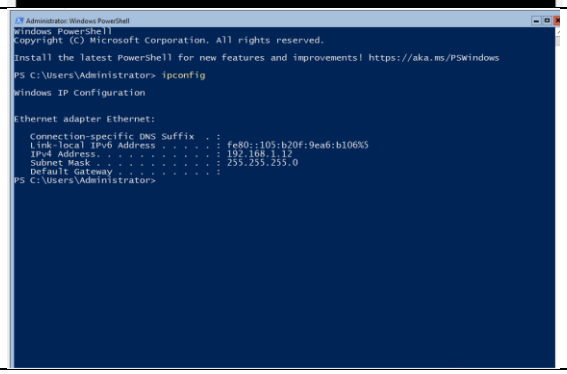
1st Login

After installing Windows Server, we log in for the first time using the local Administrator account to begin configuring the server.



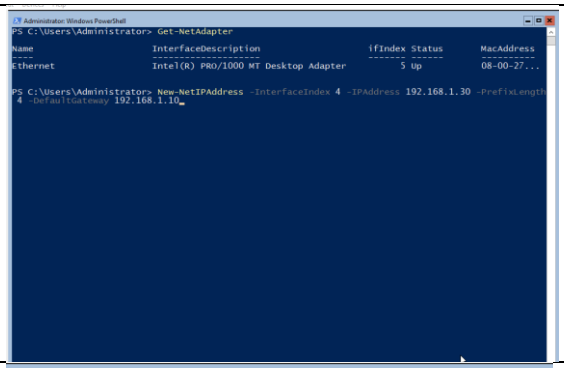
Change the Server IP Address

We assign a static IP address to the server. Static IPs are important for servers so that other devices on the network can consistently find and communicate with them.



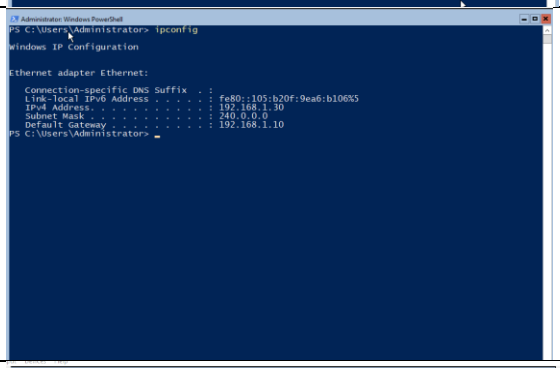
Change the Server IP Address 2

Using this exact command **New-NetIPAddress -InterfaceAlias "Ethernet" -IPAddress 192.168.1.30 -PrefixLength 24 -DefaultGateway 192.168.1.10**



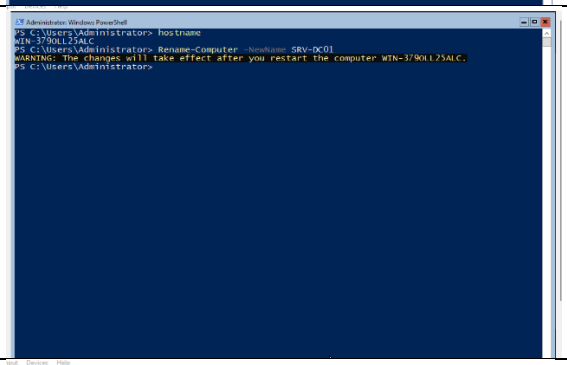
Verify IP Address is changed

We use PowerShell ipconfig to make sure our static IP and DNS settings are correctly in place.



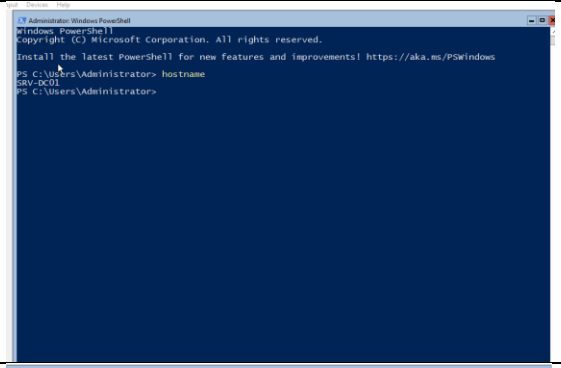
Renaming Server Name

The default name (like WIN-XXXX) is changed to SRV-DC01 to reflect its role as a domain controller.



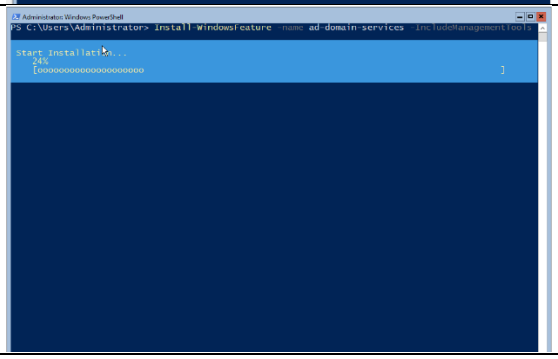
Verify Name Has Changed

After restarting the server, we confirm that the hostname has been updated using hostname



Install Active Directory Domain Services (AD DS) role

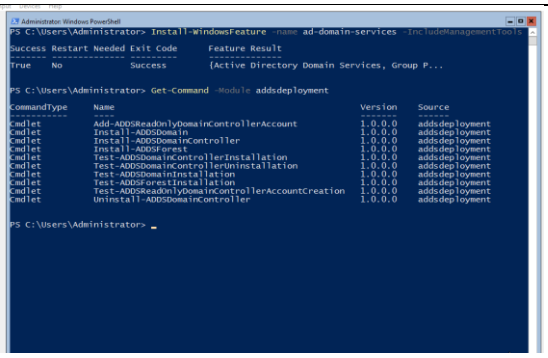
We install the AD DS role using PowerShell. This adds the necessary features for managing users, computers, and security in a domain environment.



Promote the server to a Domain Controller

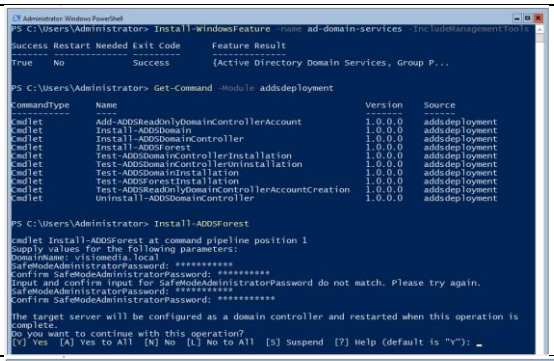
(Get-Command -Module addsdeployment)

We explore the addsdeployment module to find commands that can promote our server to a domain controller, such as Install-ADDSForest.



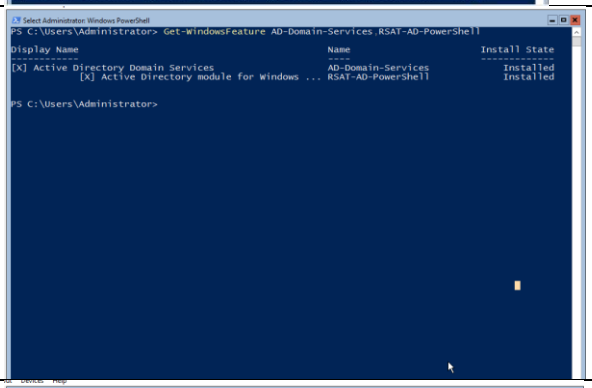
Promote the server to a Domain Controller (Install-ADDSForestInstallation)

Using PowerShell, we promote our server to become the first Domain Controller in a new forest. This is the backbone of our domain environment.



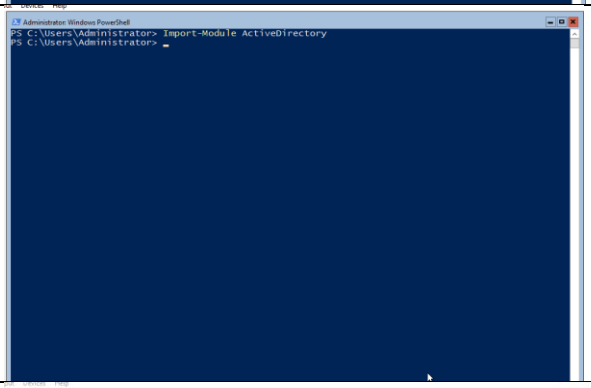
Verify Installations

We check that the domain controller promotion completed successfully by verifying DNS zones, domain info, and system logs.



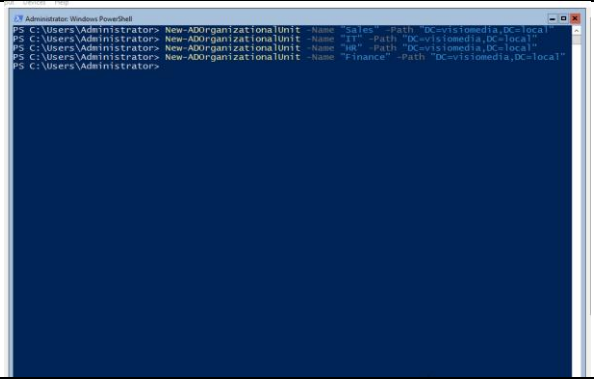
Import-Module ActiveDirectory

Before managing users and groups, we load the Active Directory module in PowerShell. This module contains the commands needed for AD management.



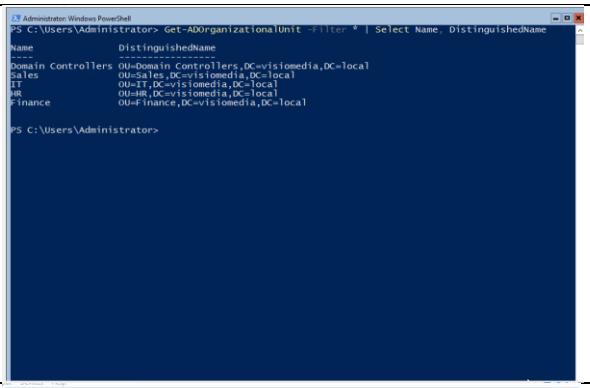
Creating Organizational Units

We use PowerShell to create Organizational Units (OUs) like "Sales", "IT", or "HR" to logically organize users and apply group policies later.



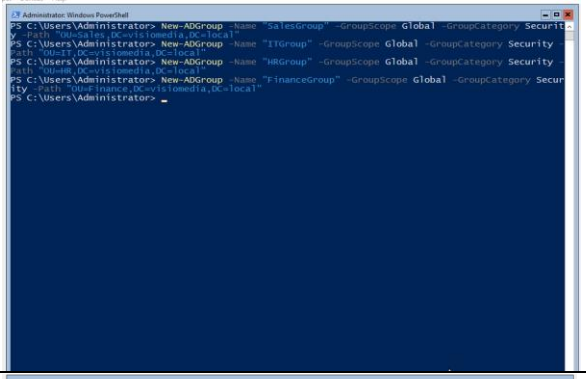
Verify that OUs have been created

We confirm the OUs exist using commands like Get-ADOrganizationalUnit or by listing the directory structure.



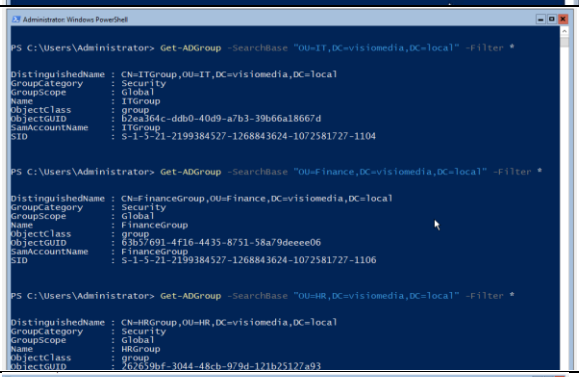
Creating Security Groups

We create Security Groups (SGs) within the OUs. These groups help manage permissions and access to network resources.



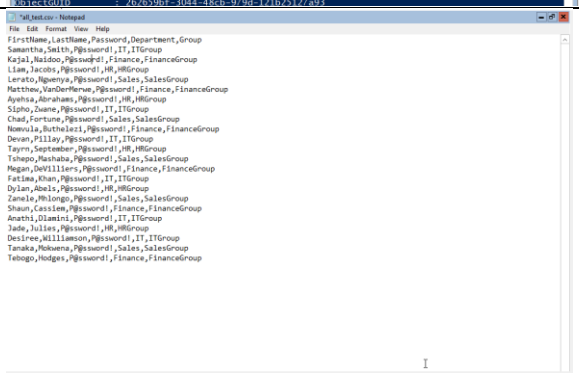
Verify that SGs have been created

We verify the groups were created using Get-ADGroup and ensure they are placed in the correct OUs.



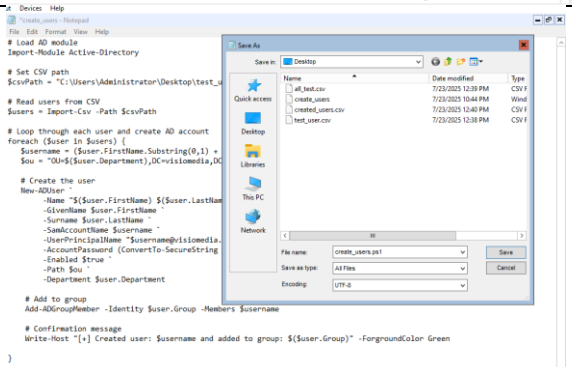
Create .csv file with users

A CSV file is created with user information such as First Name, Last Name, Password, Department and Group. This file will serve as input for our script.



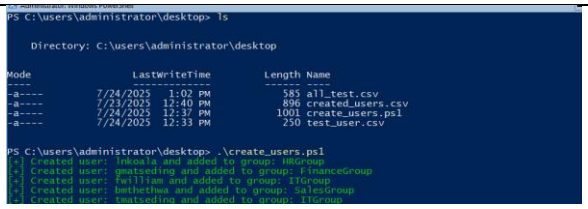
Create script that automates user creation & assign them OUs & SGs

We write a PowerShell script that reads the CSV file and automatically creates each user, placing them in the correct OU and adding them to the appropriate group.



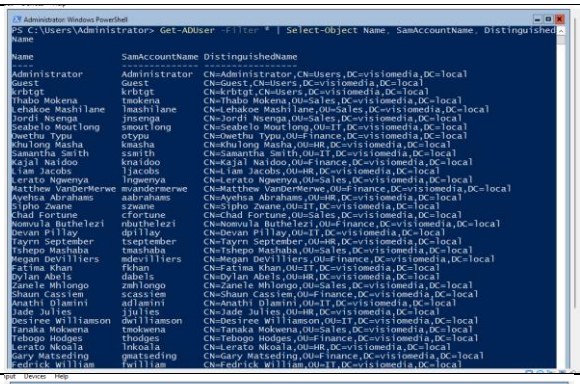
Execute Script

We run the script on the domain controller. PowerShell processes the CSV and creates all users in one go it is fast, efficient, and repeatable.



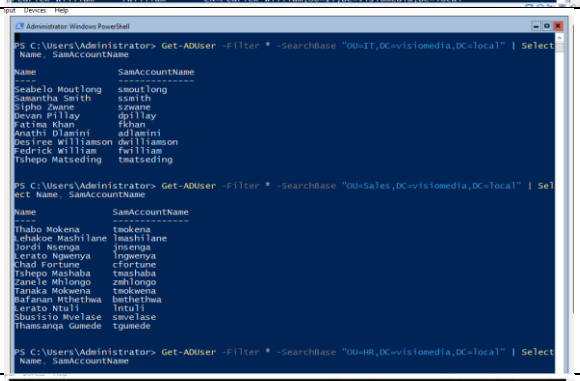
Verify users have been created

We check that the users exist using Get-ADUser, and confirm they are in the right OUs and SGs by checking their properties.



Filtering users

We use PowerShell filters (e.g., Where-Object) to search for specific users based on department, group membership, or name. This helps validate the structure.

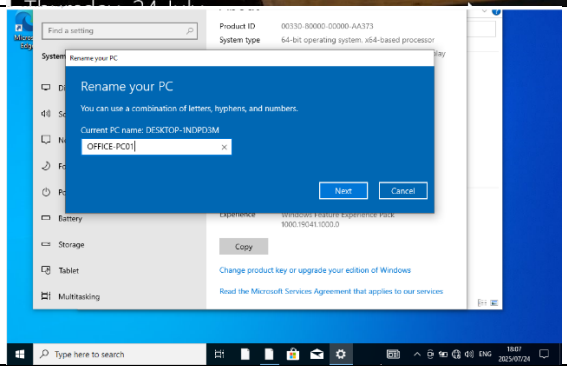


Login into Office-PC01

We now go to a client PC (Office-PC01) that will join the domain. We first log in using the local admin account.

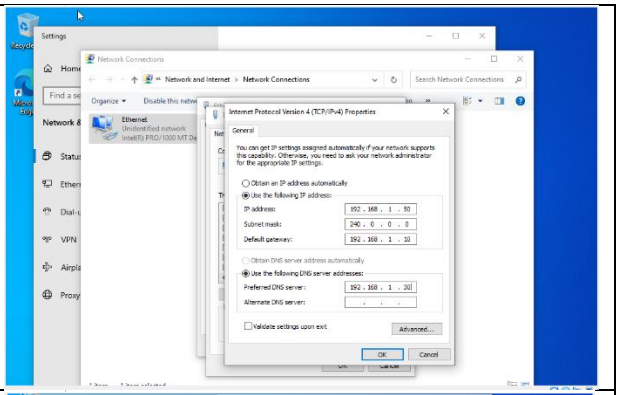
Rename Office-PC01

Just like the server, we rename the client machine to something recognizable (e.g., OfficePC01) for better network management.



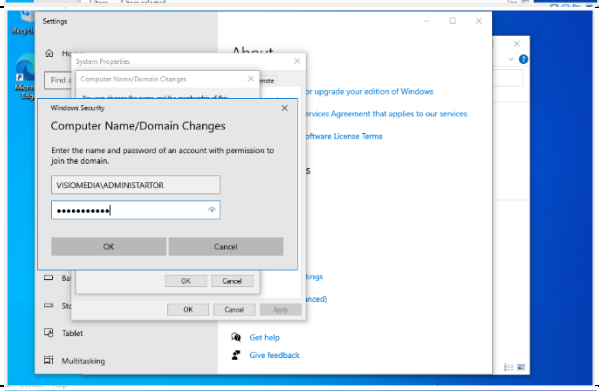
Changing IP Address and DNS

We set a static IP and assign the DNS server to the domain controller's IP address so the client can locate the domain.



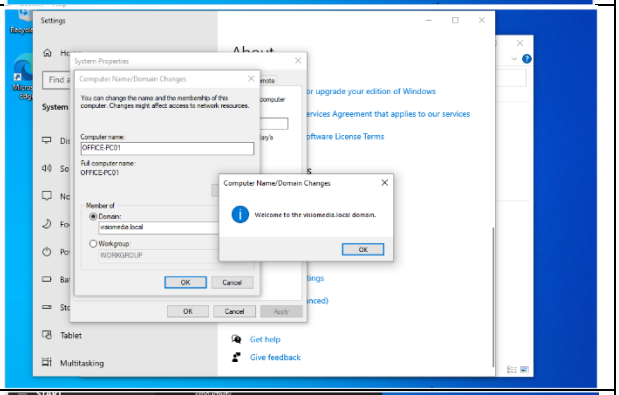
Joining Office-PC01 to domain

From Office-PC01, we joined the machine to the domain **visiomedia.local**, which is managed by the domain controller (our server at **192.168.1.30**). This step is important because it allows the client computer to become part of the domain environment and be managed centrally.



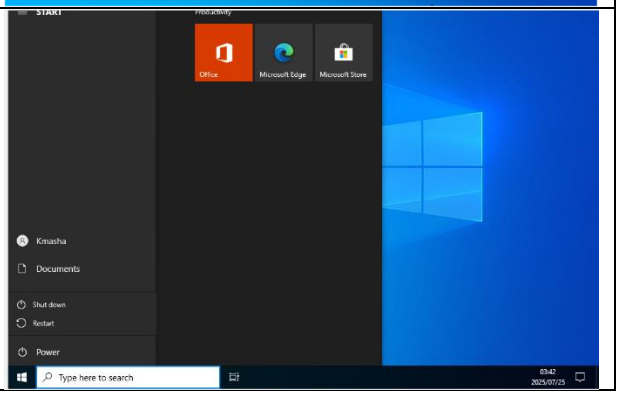
Office-PC01 joined the domain

We restart the machine and confirm that it successfully joined the domain.



Login in with user

Finally, we log in using one of the new AD user accounts created from the CSV. If everything works, we're now running in a fully domain-integrated environment.

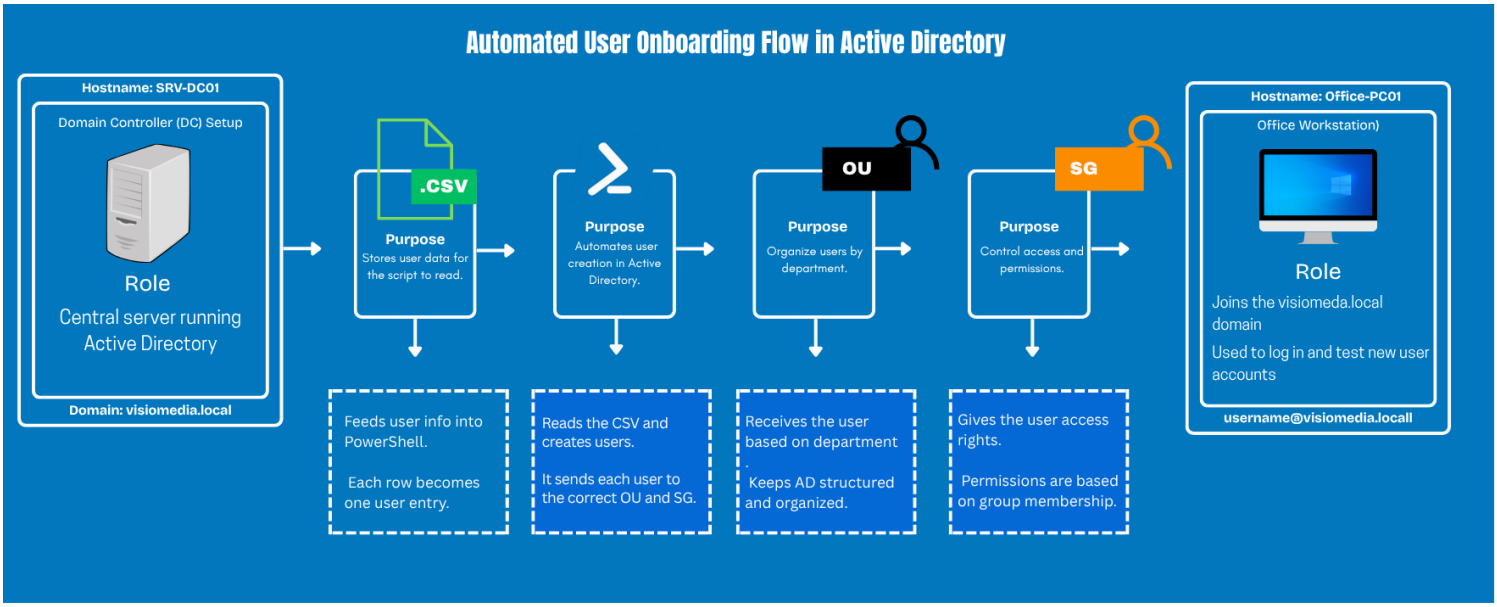


Project Summary

In this project, we set up a Windows Server 2022 environment and automated the creation of Active Directory (AD) users using PowerShell and a CSV file. We configured the domain controller, created organizational units (OUs) and security groups, and then used a script to bulk-add users and assign them to the correct groups and departments.

We also prepared a client machine (Office-PC01), configured its IP and DNS settings, and joined it to the domain directly from the PC itself. Finally, we verified that domain users could log in successfully, confirming that our automation and domain setup worked as intended.

This project demonstrates how scripting and proper AD structure can streamline user onboarding in a networked environment—an essential skill in IT administration.



Additional Improvements

Now that the core of the project is done, there are a few improvements I'd like to make to push it closer to a real-world setup:

Apply Group Policies (GPO)

I'd like to set up Group Policies to control user settings across the domain—things like enforcing strong passwords, limiting Control Panel access, or setting a default desktop wallpaper. It's a good way to maintain consistency and security.

Create Home Folders with Permissions

Another step would be to automatically create personal folders for each user on a shared drive and assign the right NTFS permissions. This is something you'd see in most companies and helps keep user files organized and private.

Automate Software Deployment

I'd also add a script or use GPO to push common apps like Chrome, 7-Zip, or antivirus software to all domain-joined PCs. It would save time and make sure every device has the tools it needs.

User Offboarding Script

As a follow-up to the onboarding script, I want to create a PowerShell script that can safely disable or remove users, unassign them from groups, and archive their folders when they leave the company.

Add Basic Security Auditing

Lastly, I'd enable basic security monitoring—like logging failed login attempts or tracking user activity with auditpol or Event Viewer. It's important for knowing what's happening on the network and responding to potential issues.