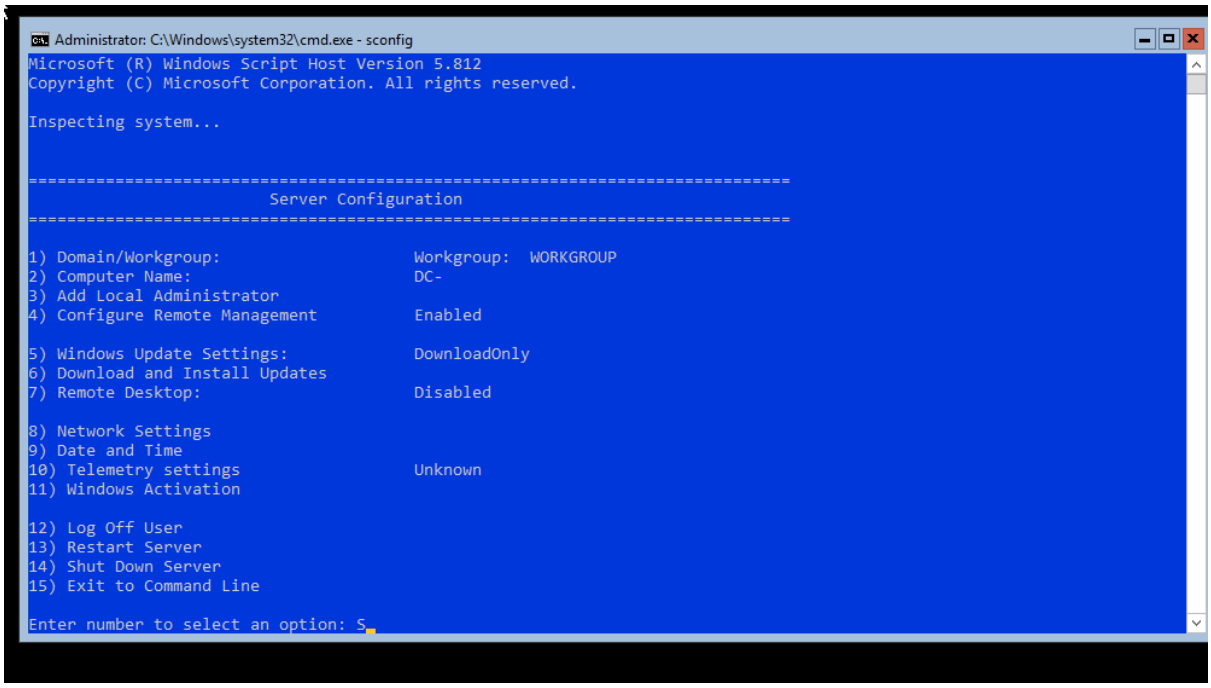
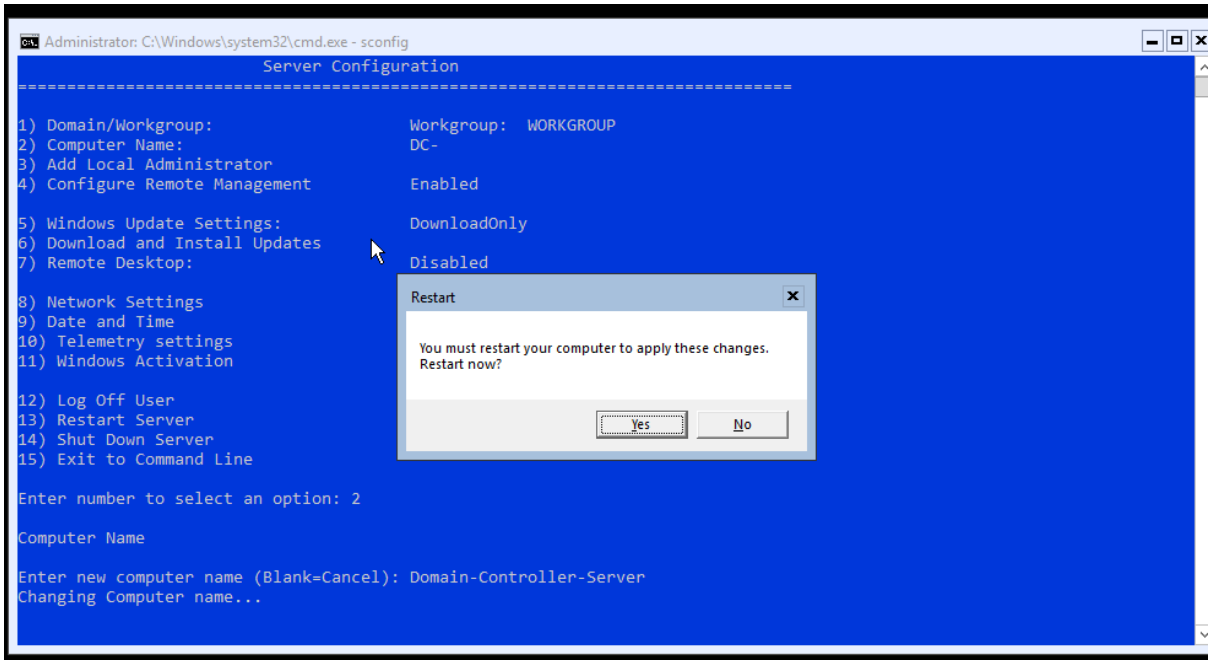
Windows Server From Nothing

Edition 2

Upon installing the Windows Server 2019 ISO from Microsoft, I've used ***sconfig*** and opened the server configuration menu.

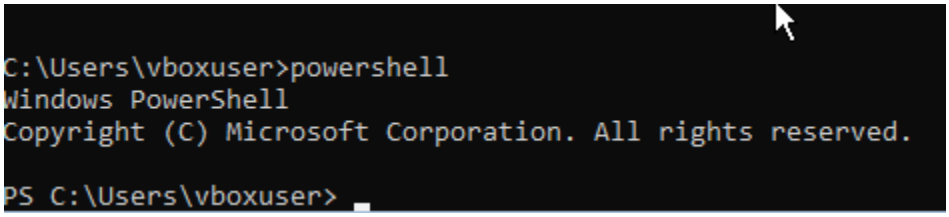


It looks simple in theory but makes me feel like a genius. I have started tinkering with the settings and changed the computer name too.

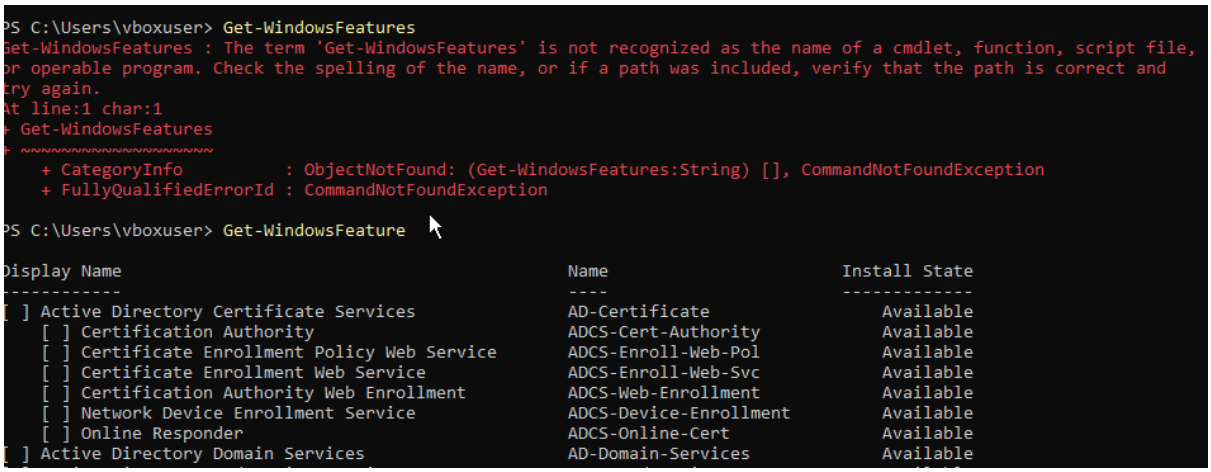


I realised, more or less after installing it, that this is the Windows Server *Core*, and not the desktop experience. I did some online research, and learned that it should, in theory, be possible to administer active directory through the command line interface alone. It should be a fun challenge, most definitely difficult, but I should learn a great deal.

I have gotten to the PowerShell interface by simply typing ***powershell*** into the command module, which makes it easy.



AI suggests that I install Active Directory Domain Services now using some cryptic powershell command, but since I do not know how to use that command yet, I've used the ***Get*** command to understand what features I can actually add to the Windows Server. This'll make it handy since it is basically a catalogue of all windows features that I might ever need in administering or developing this Domain Controller.



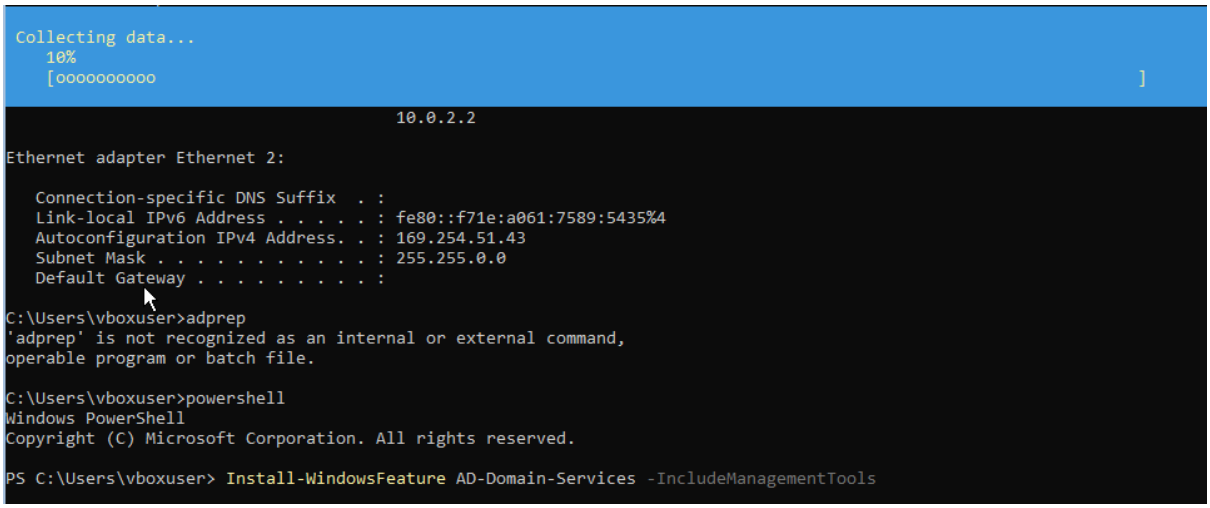
By the way, what is a domain controller anyway? It always sounded so fancy and authoritative whenever I heard it. Here's a nice online source that talks about it: "*A domain controller is the server responsible for managing network and identity security requests. It acts as a gatekeeper and authenticates whether the user is authorized to access the IT resources in the domain*" (SolarWinds 2025). Isn't that neat?

I never realized it was a security thing, but it makes sense, you wouldn't want random users on your network accessing your resources.

Now, I need to install active directory domain services on my Windows Server Core, Microsoft provides some documentation on how to do this, as well as the powershell lines themselves, which is handy.

https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/deploy/install-active-directory-domain-services--level-100

It's amusing how easy that was, especially considering I didn't have to craft the command myself. Microsoft describes this as "AD DS server role and installs the AD DS and Active Directory Lightweight Directory Services (AD LDS) server administration tools, including GUI-based tools such as Active Directory Users and Computers and command-line tools such as dcdia.exe". They note how server administration tools are not installed by default, but I don't understand the use case of installing ADDS, but not server admin tools.



The command used above was:

Install-WindowsFeature -name AD-Domain-Services -IncludeManagementTools

```
PS C:\Users\vboxuser> Get-Command -Module ADDSDeployment

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Cmdlet          Add-ADDSReadOnlyDomainControllerAccount            1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSDomain                                 1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSDomainController                       1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSForest                                 1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainControllerInstallation              1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainControllerUninstallation            1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainInstallation                        1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSForestInstallation                        1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSReadOnlyDomainControllerAccountCreation   1.0.0.0    ADDSDeployment
Cmdlet          Uninstall-ADDSDomainController                     1.0.0.0    ADDSDeployment
```

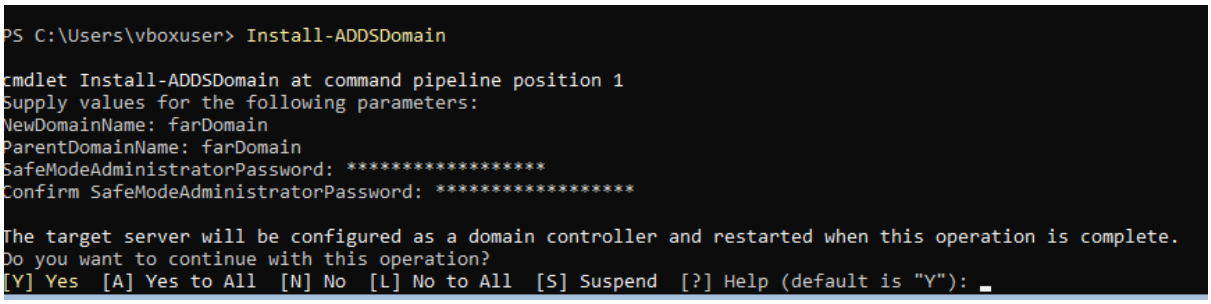
This command:

Get-Command -Module ADDSDeployment

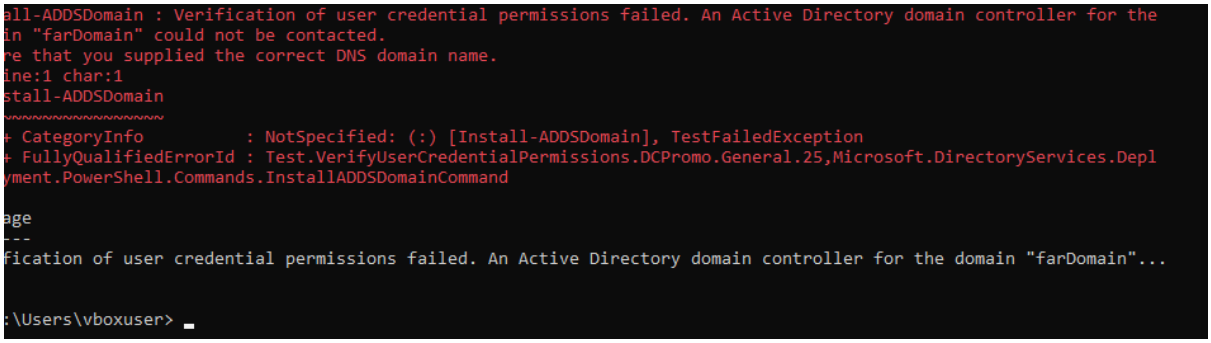
Which is listed by Microsoft, gets the available cmdlets in the ADDSDeployment module, which will be handy if I need them. Which they are, now that I need to setup a domain.

```
CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Cmdlet          Add-ADDSReadOnlyDomainControllerAccount            1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSDomain                                 1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSDomainController                       1.0.0.0    ADDSDeployment
Cmdlet          Install-ADDSForest                                 1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainControllerInstallation              1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainControllerUninstallation            1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSDomainInstallation                        1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSForestInstallation                        1.0.0.0    ADDSDeployment
Cmdlet          Test-ADDSReadOnlyDomainControllerAccountCreation   1.0.0.0    ADDSDeployment
Cmdlet          Uninstall-ADDSDomainController                     1.0.0.0    ADDSDeployment


PS C:\Users\vboxuser> Install-ADDSDomain

cmdlet Install-ADDSDomain at command pipeline position 1
Supply values for the following parameters:
NewDomainName: farDomain
ParentDomainName: farDomain
SafeModeAdministratorPassword: ******************
```

In my heart, I truly feel that setting up a domain before setting up a domain controller is wise.



Entirely confused by this statement, it wants to turn my domain into a domain controller.



Ah yes, big old red text. I really enjoy seeing this whenever I'm learning something, because it means I've hit my first roadblock. Clearly, whatever I did was wrong, or did not make sense, but in this case maybe I didn't do things in the right order. The error message describes that a domain controller for the domain 'farDomain' (which is what I wanted to name my domain), could not be contacted.

ChatGPT asks that I set my IP address on the machine to static first, and describes its reasoning as ADDS needing to rely on DNS to locate domain controllers and other services. Basically, if the IP of the DC changes, other machines and the DC itself wouldn't be able to resolve domain names in the domain, you'd think the DC could at least find itself, but I guess not.

But it makes sense for other services, DCs are the cornerstone of ADDS from what I've seen, they're needed for every directory-wide policy or anything such, and to authenticate users. It's clearly important, so it makes sense it should be easily found.

```
Select (D)HCP, (S)tatic IP (Blank=Cancel): S

Set Static IP
Enter static IP address: 192.168.1.50
Enter subnet mask (Blank = Default 255.255.255.0):
Enter default gateway: 192.168.1.1
Setting NIC to static IP...
```

Guess this means I've set it up correctly, at least I've done everything it's asked of me. I understand the need for a static IP address, and I understand the subnet too. The AI describes the use of the default gateway, which is used for handling traffic outside the local network. It needs it for things like software updates, time sync, and external DNS lookups (which I don't plan to make my server do much of). This is a pretty cool summary of those things:

IP Address: "Where am I?"

Subnet Mask: "Who is local?"

Gateway: "Where do I send traffic outside?"

DNS: "Who answers name lookups for the domain?"

Also, since it needs to be able to find itself, I've set the DNS to itself.

```
Select option:  2
DNS Servers

Enter new preferred DNS server (Blank=Cancel): 192.168.1.50
Enter alternate DNS server (Blank = none):
```

Sconfig is proving invaluable honestly, it's not really a wizard, but it is quite magical indeed.

Hopefully it works this time.

```
PS C:\Users\vboxuser> install-addsforest -domainname far.domain.com
SafeModeAdministratorPassword: ******************
Confirm SafeModeAdministratorPassword: ******************

The target server will be configured as a domain controller and restarted when this operation is
complete.
Do you want to continue with this operation?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): y
```

It worked, I now have my very own forest, and domain.

```
PS C:\Users\vboxuser> Get-ADDomain


AllowedDNSSuffixes                : {}
ChildDomains                      : {}
ComputersContainer                : CN=Computers,DC=far,DC=domain,DC=com
DeletedObjectsContainer           : CN=Deleted Objects,DC=far,DC=domain,DC=com
DistinguishedName                 : DC=far,DC=domain,DC=com
DNSRoot                           : far.domain.com
DomainControllersContainer        : OU=Domain Controllers,DC=far,DC=domain,DC=com
DomainMode                        : Windows2016Domain
DomainSID                         : S-1-5-21-2005061878-36205755-715187568
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=far,DC=domain,DC=com
Forest                            : far.domain.com
InfrastructureMaster              : Domain-Controller-Server.far.domain.com
LastLogonReplicationInterval      :
LinkedGroupPolicyObjects          : {CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=far,DC=domain,
                                    DC=com}
LostAndFoundContainer             : CN=LostAndFound,DC=far,DC=domain,DC=com
ManagedBy                         :
Name                              : far
NetBIOSName                       : FAR
ObjectClass                       : domainDNS
ObjectGUID                        : 093268fa-4b76-4b08-83a7-e0d4b7c72ec8
ParentDomain                      :
PDCEmulator                       : Domain-Controller-Server.far.domain.com
PublicKeyRequiredPasswordRolling  : True
QuotasContainer                   : CN=NTDS Quotas,DC=far,DC=domain,DC=com
ReadOnlyReplicaDirectoryServers   : {}
```

```
PS C:\Users\vboxuser> Get-AdForest


ApplicationPartitions : {DC=ForestDnsZones,DC=far,DC=domain,DC=com, DC=DomainDnsZones,DC=far,DC=domain,DC=com}
CrossForestReferences : {}
DomainNamingMaster    : Domain-Controller-Server.far.domain.com
Domains               : {far.domain.com}
ForestMode            : Windows2016Forest
GlobalCatalogs        : {Domain-Controller-Server.far.domain.com}
Name                  : far.domain.com
PartitionsContainer   : CN=Partitions,CN=Configuration,DC=far,DC=domain,DC=com
RootDomain            : far.domain.com
SchemaMaster          : Domain-Controller-Server.far.domain.com
Sites                 : {Default-First-Site-Name}
SPNSuffixes           : {}
UPNSuffixes           : {}
```

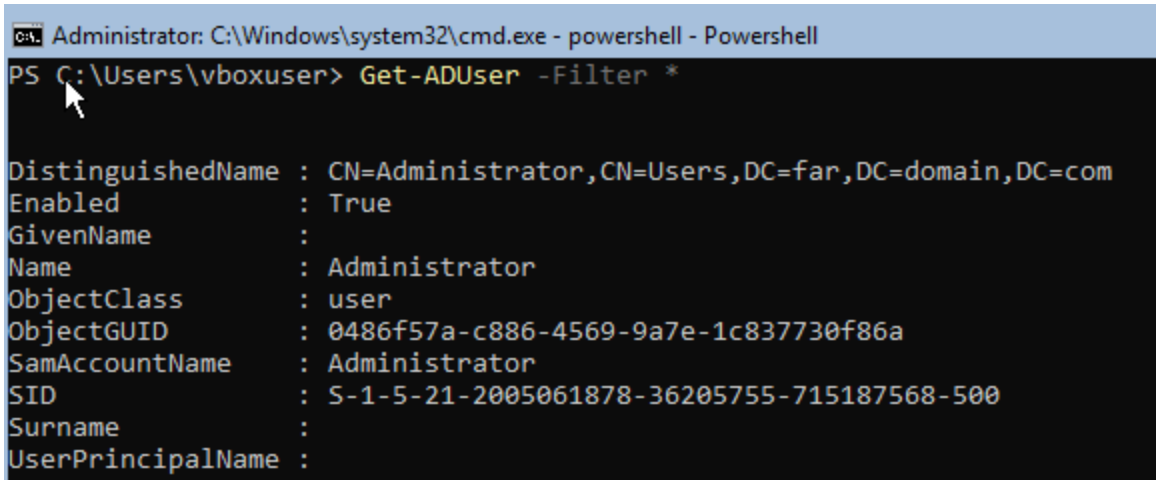
I found the syntax for creating users using the New-ADer cmdlet.

```
PS C:\Users\vboxuser> Get-Help New-AdUser

NAME
    New-ADUser

SYNTAX
    New-ADUser [-Name] <string> [-WhatIf] [-Confirm] [-AccountExpirationDate <datetime>] [-AccountNotDelegated <bool>]
    [-AccountPassword <securestring>] [-AllowReversiblePasswordEncryption <bool>] [-AuthenticationPolicy
    <ADAuthenticationPolicy>] [-AuthenticationPolicySilo <ADAuthenticationPolicySilo>] [-AuthType {Negotiate | Basic}]
    [-CannotChangePassword <bool>] [-Certificates <X509Certificate[]>] [-ChangePasswordAtLogon <bool>] [-City
    <string>] [-Company <string>] [-CompoundIdentitySupported <bool>] [-Country <string>] [-Credential <pscredential>]
    [-Department <string>] [-Description <string>] [-DisplayName <string>] [-Division <string>] [-EmailAddress
    <string>] [-EmployeeID <string>] [-EmployeeNumber <string>] [-Enabled <bool>] [-Fax <string>] [-GivenName
    <string>] [-HomeDirectory <string>] [-HomeDrive <string>] [-HomePage <string>] [-HomePhone <string>] [-Initials
    <string>] [-Instance <ADUser>] [-KerberosEncryptionType {None | DES | RC4 | AES128 | AES256}] [-LogonWorkstations
    <string>] [-Manager <ADUser>] [-MobilePhone <string>] [-Office <string>] [-OfficePhone <string>] [-Organization
    <string>] [-OtherAttributes <hashtable>] [-OtherName <string>] [-PassThru] [-PasswordNeverExpires <bool>]
    [-PasswordNotRequired <bool>] [-Path <string>] [-POBox <string>] [-PostalCode <string>]
    [-PrincipalsAllowedToDelegateToAccount <ADPrincipal[]>] [-ProfilePath <string>] [-SamAccountName <string>]
    [-ScriptPath <string>] [-Server <string>] [-ServicePrincipalNames <string[]>] [-SmartcardLogonRequired <bool>]
    [-State <string>] [-StreetAddress <string>] [-Surname <string>] [-Title <string>] [-TrustedForDelegation <bool>]
    [-Type <string>] [-UserPrincipalName <string>]  [<CommonParameters>]


ALIASES
```

This is an extremely important command, it basically informs me on everything I can do within AD, to my knowledge.

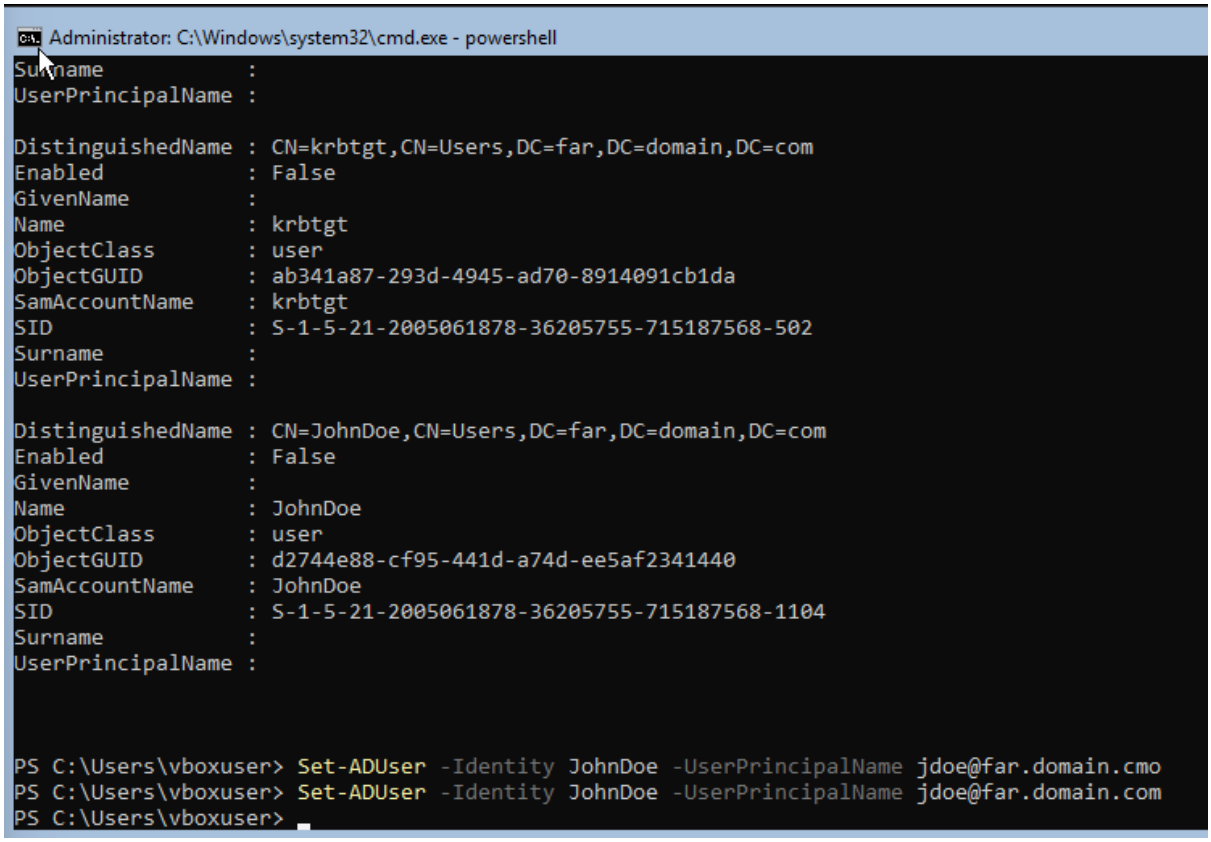
```
PS C:\Users\vboxuser> Get-Command -Module ActiveDirectory

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Cmdlet          Add-ADCentralAccessPolicyMember                    1.0.1.0    ActiveDirectory
Cmdlet          Add-ADComputerServiceAccount                       1.0.1.0    ActiveDirectory
Cmdlet          Add-ADDomainControllerPasswordReplicationPolicy    1.0.1.0    ActiveDirectory
Cmdlet          Add-ADFineGrainedPasswordPolicySubject             1.0.1.0    ActiveDirectory
Cmdlet          Add-ADGroupMember                                  1.0.1.0    ActiveDirectory
Cmdlet          Add-ADPrincipalGroupMembership                     1.0.1.0    ActiveDirectory
Cmdlet          Add-ADResourcePropertyListMember                   1.0.1.0    ActiveDirectory
Cmdlet          Clear-ADAccountExpiration                          1.0.1.0    ActiveDirectory
Cmdlet          Clear-ADClaimTransformLink                         1.0.1.0    ActiveDirectory
Cmdlet          Disable-ADAccount                                  1.0.1.0    ActiveDirectory
Cmdlet          Disable-ADOptionalFeature                          1.0.1.0    ActiveDirectory
```

It is quite crucial that I am able to view all my users, and I can do this with the following:



Edition 2

Today, I have started tinkering with the Active Directory users, and I've learned how to set userPrincipalNames and other ADUser properties using the Set-ADUser command.



We can check if the change was successful by doing the following:

```
PS C:\Users\vboxuser> Get-ADUser -Filter *
```

Since we only have a few users, we can get away with doing this for now, but as the userbase expands, we will need to start using filters to locate particular users.

```
DistinguishedName : CN=JohnDoe,CN=Users,DC=far,DC=domain,DC=com
Enabled           : False
GivenName         :
Name              : JohnDoe
ObjectClass       : user
ObjectGUID        : d2744e88-cf95-441d-a74d-ee5af2341440
SamAccountName    : JohnDoe
SID               : S-1-5-21-2005061878-36205755-715187568-1104
Surname           :
UserPrincipalName : jdoe@far.domain.com
```

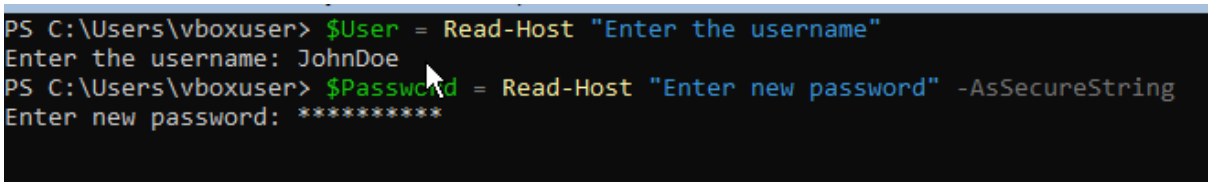
The change went through, and the userprincipalname has now been set. It is important to note that this user account is still disabled, because it doesn't have a password set, so as a system administrator I need to set this account password and enable it. I will attempt to do this using the following command:

```
PS C:\Users\vboxuser> Set-ADAccountPassword -Identity "JohnDoe" -Reset -NewPassword (ConvertTo-Se
cureString "Password!#@1012" -AsPlainText -Force)
PS C:\Users\vboxuser>
```

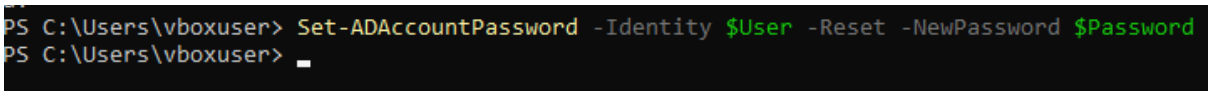
Without the 'reset' parameter, PS assumes that we want to change the password, rather than set the password. Apparently, this is what admins normally do.

AD cmdlets enforce security, so passwords cannot be passed as plain text, in this case it is wrapped in ConvertTo-SecureString, and we force it to go through, since AD will warn us that using plain text is not secure. In a circumstance where we have an established key vault, Windows Credential Manager, SecretManagement modules, etc. For this test, this is okay for our fictional user.

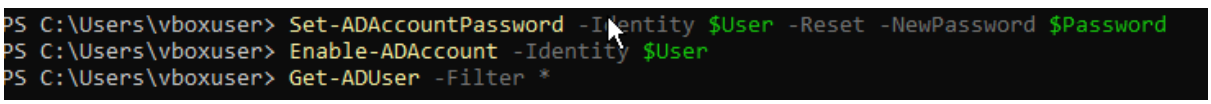
There is another, more secure way to do it, where it doesn't show up in powershell history. It apparently doesn't even show up in powershell, which I'm going to test now. We can actually store things in memory securely, to use them in the powershell command later. In Java or Python, this would basically be using variables to accomplish this task. Not sure what it's called in PS.



It looks like using 'variables', or whatever they're called in PS, has actually worked, not only was I able to type in the password securely, but I was also able to input those into the Set-ADAccountPassword fields.



Now, we can go ahead and enable the ADAccount.



```
DistinguishedName : CN=JohnDoe,CN=Users,DC=far,DC=domain,DC=com
Enabled           : True
GivenName         :
Name              : JohnDoe
ObjectClass       : user
ObjectGUID        : d2744e88-cf95-441d-a74d-ee5af2341440
SamAccountName    : JohnDoe
SID               : S-1-5-21-2005061878-36205755-715187568-1104
Surname           :
UserPrincipalName : jdoe@far.domain.com
```

The next thing to do is create groups, using groups we can administer users as a group, rather than individually. This is simple to do, thanks to Microsoft's documentation.

```
PS C:\users\vboxuser> New-ADGroup -Name "DCAdmin" -SamAccountName DCAdmin -GroupCategory Security


cmdlet New-ADGroup at command pipeline position 1
Supply values for the following parameters:
GroupScope: Global
PS C:\users\vboxuser> Set-ADGroup -Identity DCAdmin -description DC Administrators belong to this
 group.
Set-ADGroup : A positional parameter cannot be found that accepts argument 'Administrators'.
At line:1 char:1
+ Set-ADGroup -Identity DCAdmin -description DC Administrators belong t ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidArgument: (:) [Set-ADGroup], ParameterBindingException
    + FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.ActiveDirectory.Management.
   Commands.SetADGroup

PS C:\users\vboxuser> Set-ADGroup -Identity DCAdmin -description "DC Administrators belong to thi
s group."
PS C:\users\vboxuser> _
```

https://learn.microsoft.com/en-us/powershell/module/activedirectory/new-adgroup?view=windowsserver2025-ps

At this stage, we have a domain, domain controller, users, and a group. I have also gone ahead and replicated this and made multiple groups.

```
DistinguishedName : CN=DCAdmin,CN=Users,DC=far,DC=domain,DC=com
GroupCategory     : Security
GroupScope        : Global
Name              : DCAdmin
ObjectClass       : group
ObjectGUID        : 400a9b9c-e1b7-424d-9556-9feac3479c0d
SamAccountName    : DCAdmin
SID               : S-1-5-21-2005061878-36205755-715187568-1106

DistinguishedName : CN=HR_Users,CN=Users,DC=far,DC=domain,DC=com
GroupCategory     : Security
GroupScope        : Global
Name              : HR_Users
ObjectClass       : group
ObjectGUID        : 2a240d99-ed46-409d-a777-2de45fa4e1a6
SamAccountName    : HRUser
SID               : S-1-5-21-2005061878-36205755-715187568-1107

DistinguishedName : CN=Finance_Admins,CN=Users,DC=far,DC=domain,DC=com
GroupCategory     : Security
GroupScope        : Global
Name              : Finance_Admins
ObjectClass       : group
ObjectGUID        : 427f4859-2ad6-4b43-afe1-ed1520d3c1ae
SamAccountName    : FinanceAdmin
SID               : S-1-5-21-2005061878-36205755-715187568-1108
```

At this stage, we should add users to groups, which will make it easier to administer them together rather than must administer them individually. I've gone ahead and added a member to the Finance Admin group created.

```
PS C:\users\vboxuser> Add-ADGroupMember -Identity Finance_Admins -Members janeDoe
Add-ADGroupMember : Cannot find an object with identity: 'Finance_Admins' under:
'DC=far,DC=domain,DC=com'.
At line:1 char:1
+ Add-ADGroupMember -Identity Finance_Admins -Members janeDoe
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (Finance_Admins:ADGroup) [Add-ADGroupMember], ADI
   dentityNotFoundException
    + FullyQualifiedErrorId : ActiveDirectoryCmdlet:Microsoft.ActiveDirectory.Management.ADIden
   tityNotFoundException,Microsoft.ActiveDirectory.Management.Commands.AddADGroupMember

PS C:\users\vboxuser> Add-ADGroupMember -Identity FinanceAdmin -Members janeDoe
PS C:\users\vboxuser> Get-ADGroupMember -Identity FinanceAdmin


distinguishedName : CN=Jane Doe,CN=Users,DC=far,DC=domain,DC=com
name              : Jane Doe
objectClass       : user
objectGUID        : 91499cec-2251-4977-849a-23d074feb26d
SamAccountName    : janeDoe
SID               : S-1-5-21-2005061878-36205755-715187568-1109
```

I realise now, through experimenting, that the SamAccountName is basically an object identifier, rather than the actual name, when dealing with putting users into groups and stuff like that. Something like the barcode of a product, as opposed to the actual name of the product, or something like that.

https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/deploy/install-active-directory-domain-services--level-100-#install-ad-ds-by-using-windows-powershell