Sign in

lkys37en / **Start-ADEnum** Public

Notifications | Fork 19 | Star 88

Code | Issues 2 | Pull requests | Actions | Projects | Security | Insights

Start-ADEnum / Functions / **Start-ADEnum.ps1**

845 lines (715 loc) · 48.1 KB

Code | Blame | Raw

```
Function Start-ADEnum {
    <#
    .SYNOPSIS
    Author: @lkys37en
    Required Dependencies: Powershell Version 5.x, Windows 10 1803 or 1809
    Optional Dependencies: None

    .DESCRIPTION
    This tool is used to automate Active Directory enumeration. The tool requires a domain context

    .PARAMETER ClientName
    Enter clientname for folder structure.

    .PARAMETER Path
    Enter path where evidence will be placed. If folder doesn't already exist, the script will crea

    .PARAMETER Domains
    Enter individual domains to enumerate or let the script automatically identify all vailable dom

    .PARAMETER Scan
    Enter individual scan(s) to perform.

    .EXAMPLE
    PS C:\> Start-ADEnum  -ClientName lkylabs -Path C:\Projects -Scan All
    Collects a list of all domain/forest trusts and runs all scans against each domain found.

```

```powershell
        .EXAMPLE
        PS C:\> Start-ADEnum  -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com  -Scan All
        Runs all scans against a single domain.

        .EXAMPLE
        PS C:\> Start-ADEnum  -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com,corp.lkylabs.co
        Runs PowerView and Bloodhound scans against domains lkylabs.com and corp.lkylabs.com.

        #>
        [CmdletBinding()]
        Param (
            [Parameter(Mandatory = $True)]
            [ValidateNotNullOrEmpty()]
            [String]
            $ClientName,

            [Parameter(Mandatory = $True)]
            [ValidateNotNullOrEmpty()]
            [String]
            $Path,

            [Parameter(Mandatory = $False)]
            [String[]]
            $Domains,

            [Parameter(Mandatory = $True)]
            [ValidateSet("ADCS", "Bloodhound", "GPOReport", "PowerView", "PingCastle", "PowerUPSQL", "
            [String[]]
            $Scan,

            [Parameter(Mandatory = $False)]
            [String]
            $UserName
        )

        Begin {
            #Folders variable
            $Folders = @(
                "PingCastle"
                "PowerView"
                "Bloodhound"
                "GPO"
                "Microsoft Services\Exchange"
                "Microsoft Services\ADCS"
                "PowerUPSQL"
                "QuickScan"
```

```
        )

        #Installs all prereqs if missing
        Write-Host -ForegroundColor Green "[*] Performing prereqs check"
        Start-PrereqCheck

        Import-Module C:\Tools\PowerSploit\Recon\PowerView.ps1

        #Checking Domain Context before moving forward
        try {
            [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain() | Out-Null
        }

        catch {
            throw "[*] Not currently assoicated with a domain account, perform runas /netonly befor
        }

        #Creating Path and evidence folder structure
        if ((Test-Path $Path) -eq $false) {
            try {
                Write-Host -ForegroundColor Green "[+] Creating $Path Folder "
                mkdir -Path $Path | Out-Null
            }

            catch {
                throw "An error has occurred $($_.Exception.Message)"
            }


            try {
                Write-Host -ForegroundColor Green "[+] Creating $ClientName Folder"
                mkdir -Path $Path\$ClientName | Out-Null
            }

            catch {
                throw "An error has occurred  $($_.Exception.Message)"
            }

            try {
                if (!($Domains)) {
                    Write-Host -ForegroundColor Green "[*] Collecting a list of domains.."
                    $Domains = (Get-DomainTrustMapping -API).TargetName | Select-Object -Unique
                    if ($Domains -eq $null) { $Domains = (Get-NetDomain).Name }
                }

                foreach ($Domain in $Domains) {
```

```
                        foreach ($Domain in $Domains) {
```

```
                "All" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting All AD Enum for $Domain"
                        Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Path,
                        Start-Job -ScriptBlock $PowerUPSQLScriptBlock -ArgumentList $ClientName, $Path,
                        Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Path,
                        Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Path,
                        Start-Job -ScriptBlock $GPOReportScriptBlock -ArgumentList $ClientName, $Path,
                        Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Path, $D
                        Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Path, $Domain -Name AD
                    }
                }

                "ADCS" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting AD Certificate Services Enum f
                        Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Path, $Domain -Name AD
                    }
                }

                "PowerView" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting PowerView Enum for $Domain"
                        Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Path,
                    }
                }

                "PowerUPSQL" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting PowerUPSQL Enum for $Domain"
                        Start-Job -ScriptBlock $PowerUPSQLScriptBlock -ArgumentList $ClientName, $Path,
                    }
                }
```

```
                "PingCastle" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting Ping Castle AD Enum for $Domai
                        Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Path,
                    }
                }

                "Bloodhound" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting Bloodhound AD Enum for $Domain"
                        Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Path,
                    }
                }

                "GPOReport" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting GPO Report AD Enum for $Domain"
                        Start-Job -ScriptBlock $GPOReportScriptBlock -ArgumentList $ClientName, $Path,
                    }
                }

                "PrivExchange" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting PrivExchange AD Enum for $Domai
                        Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Path, $D
                    }
                }

                "QuickScan" {
                    foreach ($Domain in $Domains) {
                        Write-Host -ForegroundColor Green "[+] Starting QuickScan PingCastle AD Enum f
                        Start-Job -ScriptBlock $QuickScanPingCastle -ArgumentList $ClientName, $Path, $

                        Write-Host -ForegroundColor Green "[+] Starting QuickScan PowerUPSQL AD Enum f
                        Start-Job -ScriptBlock $QuickScanMSSQL -ArgumentList $ClientName, $Path, $Domai
                    }
                }
            }
        }
    }
```