





Product Solutions Resources Open Source Enterprise Pricing



Sign in

Sign up

lkys37en / Start-ADEnum 

Public

Notifications

Fork 19

Star 88

Code

Issues 2

Pull requests

Actions

Projects

Security

Insights

Files

5b42c54



Go to file

▼

Functions

Install-Tools.ps1

Start-ADEnum.ps1

Start-Prereqcheck.ps1

LICENSE

README.md

Start-ADEnum.psd1

Start-ADEnum.psm1

Start-ADEnum / Functions / Start-ADEnum.ps1



History

Code

Blame

845 lines (715 loc) · 48.1 KB

Raw







1Function Start-ADEnum {

2<

3.SYNOPSIS

4Author: @lkys37en

5Required Dependencies: Powershell Version 5.x, Windows 10 1803 or 1809

6Optional Dependencies: None

7

8.DESRIPTION

9This tool is used to automate Active Directory enumeration. The tool requires a dom

10

11.PARAMETER ClientName

12Enter clientname for folder structure.

13

14.PARAMETER Path

15Enter path where evidence will be placed. If folder doesn't already exist, the scri

16

17.PARAMETER Domains

18Enter individual domains to enumerate or let the script automatically identify all

19

20.PARAMETER Scan

21Enter individual scan(s) to perform.

22

23.EXAMPLE

24PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Scan All

25Collects a list of all domain/forest trusts and runs all scans against each domain

26

27.EXAMPLE

28PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com -S

29Runs all scans against a single domain.

30

31.EXAMPLE

32PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com,cor

33Runs PowerView and Bloodhound scans against domains lkylabs.com and corp.lkylabs.co

34

35#>

36[CmdletBinding()]

37Param (

38[Parameter(Mandatory = \$True)]

39[ValidateNotNullOrEmpty()]

40[String]

41\$ClientName,

42

43[Parameter(Mandatory = \$True)]

44[ValidateNotNullOrEmpty()]

45[String]

46\$Path,

47

48[Parameter(Mandatory = \$False)]

49[String[]]

50\$Domains,

51

52[Parameter(Mandatory = \$True)]

53[ValidateSet("ADCS", "Bloodhound", "GPOReport", "PowerView", "PingCastle", "Pow

54[String[]]

55\$Scan,

56

57[Parameter(Mandatory = \$False)]

Page 1 of 12

```
57         [Parameter(Mandatory = $false)]
58         [String]
59         $UserName
60     )
61
62     Begin {
63         #Folders variable
64         $Folders = @(
65             "PingCastle"
66             "PowerView"
67             "Bloodhound"
68             "GPO"
69             "Microsoft Services\Exchange"
70             "Microsoft Services\ADCS"
71             "PowerUPSQL"
72             "QuickScan"
73         )
74
75         #Installs all prereqs if missing
76         Write-Host -ForegroundColor Green "[*] Performing prereqs check"
77         Start-PrereqCheck
78
79         Import-Module C:\Tools\PowerSploit\Recon\PowerView.ps1
80
81         #Checking Domain Context before moving forward
82         try {
83             [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain() | Out-Null
84         }
85
86         catch {
87             throw "[*] Not currently associated with a domain account, perform runas /n
88         }
89
90         #Creating Path and evidence folder structure
91         if ((Test-Path $Path) -eq $false) {
92             try {
93                 Write-Host -ForegroundColor Green "[+] Creating $Path Folder "
94                 mkdir -Path $Path | Out-Null
95             }
96
97             catch {
98                 throw "An error has occurred $($_.Exception.Message)"
99             }
100
101
102             try {
103                 Write-Host -ForegroundColor Green "[+] Creating $ClientName Folder"
104                 mkdir -Path $Path\$ClientName | Out-Null
105             }
106
107             catch {
108                 throw "An error has occurred $($_.Exception.Message)"
109             }
110
111             try {
112                 if (!$Domains) {
113                     Write-Host -ForegroundColor Green "[*] Collecting a list of domains
114                     $Domains = (Get-DomainTrustMapping -API).TargetName | Select-Object
115                     if ($Domains -eq $null) { $Domains = (Get-NetDomain).Name }
116                 }
117
118                 foreach ($Domain in $Domains) {
```



















```
772         "All" {
773             foreach ($Domain in $Domains) {
774                 Write-Host -ForegroundColor Green "[+] Starting All AD Enum for $Domain"
775                 Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Domain
776                 Start-Job -ScriptBlock $PowerUPSQLScriptBlock -ArgumentList $ClientName, $Domain
777                 Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Domain
778                 Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Domain
779                 Start-Job -ScriptBlock $GPOResultScriptBlock -ArgumentList $ClientName, $Domain
780                 Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Domain
781                 Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Domain
782             }
783         }
784
785         "ADCS" {
786             foreach ($Domain in $Domains) {
787                 Write-Host -ForegroundColor Green "[+] Starting AD Certificate Service Enum for $Domain"
788                 Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Domain
789             }
790         }
791
792         "PowerView" {
793             foreach ($Domain in $Domains) {
794                 Write-Host -ForegroundColor Green "[+] Starting PowerView Enum for $Domain"
795                 Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Domain
796             }
797         }
798
799         "PowerUPSQL" {
800             foreach ($Domain in $Domains) {
801                 Write-Host -ForegroundColor Green "[+] Starting PowerUPSQL Enum for $Domain"
802                 Start-Job -ScriptBlock $PowerUPSQLScriptBlock -ArgumentList $ClientName, $Domain
803             }
804         }
```

```
803         }
804     }
805
806     "PingCastle" {
807         foreach ($Domain in $Domains) {
808             Write-Host -ForegroundColor Green "[+] Starting Ping Castle AD Enum"
809             Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Domain
810         }
811     }
812
813     "Bloodhound" {
814         foreach ($Domain in $Domains) {
815             Write-Host -ForegroundColor Green "[+] Starting Bloodhound AD Enum"
816             Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Domain
817         }
818     }
819
820     "GPOReport" {
821         foreach ($Domain in $Domains) {
822             Write-Host -ForegroundColor Green "[+] Starting GPO Report AD Enum"
823             Start-Job -ScriptBlock $GPOReportScriptBlock -ArgumentList $ClientName, $Domain
824         }
825     }
826
827     "PrivExchange" {
828         foreach ($Domain in $Domains) {
829             Write-Host -ForegroundColor Green "[+] Starting PrivExchange AD Enum"
830             Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Domain
831         }
832     }
833
834     "QuickScan" {
835         foreach ($Domain in $Domains) {
836             Write-Host -ForegroundColor Green "[+] Starting QuickScan PingCastle"
837             Start-Job -ScriptBlock $QuickScanPingCastle -ArgumentList $ClientName, $Domain
838
839             Write-Host -ForegroundColor Green "[+] Starting QuickScan PowerUPSQL"
840             Start-Job -ScriptBlock $QuickScanMSSQL -ArgumentList $ClientName, $Domain
841         }
842     }
843 }
844 }
845 }
```