








Sign in


 fengjixuchui / Start-ADEnum

Public


 Notifications


 Fork


0


 Star


0


 Code


 Pull requests


 Actions

 Projects

 Security




 Insights

Start-ADEnum / Functions / Start-ADEnum.ps1 



520 lines (435 loc) · 26.9 KB

CodeBlame

Raw

```
1  Function Start-ADEnum {
2      <#
3      .SYNOPSIS
4      Author: @lkys37en
5      Required Dependencies: Powershell Version 5.x, Windows 10 1803 or 1809
6      Optional Dependencies: None
7
8      .DESCRIPTION
9      This tool is used to automate Active Directory enumeration. The tool requires a domain context
10
11     .PARAMETER ClientName
12     Enter clientname for folder structure.
13
14     .PARAMETER Path
15     Enter path where evidence will be placed. If folder doesn't already exist, the script will crea
16
17     .PARAMETER Domain
18     Enter individual domain to enumerate or let the script automatically identify all vailable doma
19
20     .PARAMETER Scan
21     Enter individual scan(s) to perform.
22
23     .EXAMPLE
24     PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Scan All
25     Collects a list of all domain/forest trusts and runs all scans against each domain found.
```

```
26
27     .EXAMPLE
28     PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com -Scan All
29     Runs all scans against lkylabs.com and corp.lkylabs.com.
30
31     .EXAMPLE
32     PS C:\> Start-ADEnum -ClientName lkylabs -Path C:\Projects -Domain lkylabs.com,corp.lkylabs.com
33     Runs PowerView and Bloodhound scans against lkylabs.com and corp.lkylabs.com domains.
34
35     #>
36     [CmdletBinding()]
37     Param (
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", "PrivExchange",
54         [String[]]
55         $Scan
56     )
57
58     Begin {
59         #Folders variable
60         $Folders = @(
61             "PingCastle"
62             "PowerView"
63             "Bloodhound"
64             "GPO"
65             "Microsoft Services\Exchange"
66             "Microsoft Services\ADCS"
67         )
68
69         #Installs all prereqs if missing
70         Write-Host -ForegroundColor Green "[*] Performing prereqs check"
71         Start-PrereqCheck
```

```
72
73     Import-Module C:\Tools\PowerSploit\Recon\PowerView.ps1
74
75     #Creating Path and evidence folder structure
76     if ((Test-Path $Path) -eq $false) {
77         try {
78             Write-Host -ForegroundColor Green "[+] Creating $Path Folder "
79             mkdir -Path $Path | Out-Null
80         }
81
82         catch {
83             throw "An error has occurred $($_.Exception.Message)"
84         }
85
86
87         try {
88             Write-Host -ForegroundColor Green "[+] Creating $ClientName Folder"
89             mkdir -Path $Path\$ClientName | Out-Null
90         }
91
92         catch {
93             throw "An error has occurred $($_.Exception.Message)"
94         }
95
96         try {
97             if ($Domain) {
98                 Write-host -ForegroundColor Green "[+] Performing Enumeration only on $Domain"
99                 $Domains = $Domain
100             }
101
102             else {
103                 $Domains = (Get-DomainTrustMapping -API).TargetName | Select-Object -Unique
104                 if ($Domains -eq $null) { $Domains = (Get-NetDomain).Name }
105             }
106
107             foreach ($Domain in $Domains) {
108                 Write-Host -ForegroundColor Magenta "[*] Creating $Domain evidence folders"
109                 mkdir -Path "$Path\$ClientName\$Domain" | Out-Null
110
111                 foreach ($folder in $folders) {
112                     mkdir -Path "$Path\$ClientName\$Domain\$folder" | Out-Null
113                 }
114             }
115         }
116
117         catch {
```

117

caten |


```
447
448     #Running all scanner commands
449     foreach ($Argument in $Arguments) {
450         Set-Location $Folder ; Start-Process C:\tools\PingCastle\PingCastle.exe -ArgumentList $Argument
451     }
452
453     $Output = (Get-ChildItem $Folder -Exclude *html*, *xml*).FullName
454
455     #Converting PingCastle text files into CSV's
456     foreach ($Item in $Output) {
457         Import-Csv -Path $Item -Delimiter "`t" | Export-Csv -Path ($Folder + ($Item -split '\.txt')[0] + ".csv")
458     }
459
460     #Remove original text files
461     (Get-ChildItem $Folder).FullName | Remove-Item -Include *.txt
462 }
463
464 switch ($Scan) {
465     "All" {
466         foreach ($Domain in $Domains) {
467             Write-Host -ForegroundColor Green "[+] Starting All AD Enum for $Domain"
468             Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Path, $Domain
469             Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Path, $Domain
470             Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Path, $Domain
471             Start-Job -ScriptBlock $GPOResultScriptBlock -ArgumentList $ClientName, $Path, $Domain
472             Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Path, $Domain
473             Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Path, $Domain -Name AD
474         }
475     }
476
477     "ADCS" {
478         foreach ($Domain in $Domains) {
479             Write-Host -ForegroundColor Green "[+] Starting AD Certificate Services Enum for $Domain"
480             Start-Job -ScriptBlock $ADCS -ArgumentList $ClientName, $Path, $Domain -Name Pr
481         }
482     }
483 }
```

```
484         "PowerView" {
485             foreach ($Domain in $Domains) {
486                 Write-Host -ForegroundColor Green "[+] Starting PowerView Enum for $Domain"
487                 Start-Job -ScriptBlock $PowerViewScriptBlock -ArgumentList $ClientName, $Path,
488             }
489         }
490
491         "PingCastle" {
492             foreach ($Domain in $Domains) {
493                 Write-Host -ForegroundColor Green "[+] Starting Ping Castle AD Enum for $Domain"
494                 Start-Job -ScriptBlock $PingCastleScriptBlock -ArgumentList $ClientName, $Path,
495             }
496         }
497
498         "Bloodhound" {
499             foreach ($Domain in $Domains) {
500                 Write-Host -ForegroundColor Green "[+] Starting Bloodhound AD Enum for $Domain"
501                 Start-Job -ScriptBlock $BloodhoundScriptBlock -ArgumentList $ClientName, $Path,
502             }
503         }
504
505         "GPOReport" {
506             foreach ($Domain in $Domains) {
507                 Write-Host -ForegroundColor Green "[+] Starting GPO Report AD Enum for $Domain"
508                 Start-Job -ScriptBlock $GPOReportScriptBlock -ArgumentList $ClientName, $Path,
509             }
510         }
511
512         "PrivExchange" {
513             foreach ($Domain in $Domains) {
514                 Write-Host -ForegroundColor Green "[+] Starting PrivExchange AD Enum for $Domain"
515                 Start-Job -ScriptBlock $PrivExchangeCheck -ArgumentList $ClientName, $Path, $Domain
516             }
517         }
518     }
519 }
520 }
```