
Product ▾Solutions ▾Resources ▾Open Source ▾Enterprise ▾Pricing

Sign in

Sign up

BloodHoundAD / BloodHoundPublic

Notifications

Fork1.7k

Star9.9k

<>Code

Issues70

Pull requests11

Actions

Projects

Wiki

Security

Insights

Files

0927441

Go to file

> .github

> Collectors

> DebugBuilds

AzureHound.ps1

SharpHound.exe

SharpHound.ps1

> docs

> src

.gitignore

LICENSE-3RD-PARTY.md

LICENSE.md

README.md

index.html

main.js

package-lock.json

package.json


renderer.js

server.js

webpack.config.development.js

webpack.config.production.js

BloodHound / Collectors / AzureHound.ps1

 andyrobbins · Fix user display name ingest error · d8163c0 · 3 years ago · History

CodeBlame1500 lines (1215 loc) · 58.2 KB

RawCopyDownloadDiff

```
1  # AzureHound Beta
2  # Authors: Andy Robbins (@wald0), Rohan Vazarkar (@cptjesus), Ryan Hausknecht (@haus3c)
3  # Copyright: SpecterOps, Inc. 2020
4
5  function Get-PrincipalMap {
6
7      $PrincipalMap = @{}
8      Get-AzureADUser -All $True | % {
9          $PrincipalMap.add($_.objectid, $_.OnPremisesSecurityIdentifier)
10     }
11     Get-AzureADGroup -All $True | % {
12         $PrincipalMap.add($_.objectid, $_.OnPremisesSecurityIdentifier)
13     }
14     $PrincipalMap
15 }
16 function Connect-AADUser {
17     $ConnectionTest = try{ [Microsoft.Open.Azure.AD.CommonLibrary.AzureSession]::Access
18     catch{"Error"}
19     If($ConnectionTest -eq 'Error'){
20         $context = [Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmProf
21         $aadToken = [Microsoft.Azure.Commands.Common.Authentication.AzureSession]::Instance
22         Connect-AzureAD -AadAccessToken $aadToken -AccountId $context.Account.Id -TenantId
23     }
24
25     function Get-AzureGraphToken
26     {
27         $APUser = Get-AzContext *>&1
28         $resource = "https://graph.microsoft.com"
29         $Token = [Microsoft.Azure.Commands.Common.Authentication.AzureSession]::Instance.Au
30         $Headers = @{}
31         $Headers.Add("Authorization","Bearer"+ " " + "$($token)")
32         $Headers
33     }
34
35     $Verbose = $True
36     function Write-Info ($Message) {
37         If ($Verbose) {
38             Write-Host $Message
39         }
40     }
41
42     function New-Output($Coll, $Type, $Directory) {
43
44         $Count = $Coll.Count
45
46         Write-Host "Writing output for $($Type)"
47         if ($null -eq $Coll) {
48             $Coll = New-Object System.Collections.ArrayList
49         }
50
51         # ConvertTo-Json consumes too much memory on larger objects, which can have million
52         # of entries in a large tenant. Write out the JSON structure a bit at a time to wor
53         # around this. This is a bit inefficient, but makes this work when the tenant becom
54         # too large.
55         $FileName = $Directory + [IO.Path]::DirectorySeparatorChar + $date + "-" + "az" + $
56         try {
57             $stream = [System.IO.StreamWriter]::new($FileName)
```

```
57         $Stream = [System.IO.StreamWriter]::new($Filename)
58
59         # Write file header JSON
60         $Stream.WriteLine('{')
61         $Stream.WriteLine("`t""meta"": {")
62         $Stream.WriteLine("`t`t""count"": $Count,")
63         $Stream.WriteLine("`t`t""type"": ""az${$Type}""")
64         $Stream.WriteLine("`t`t""version"": 4")
65         $Stream.WriteLine("`t}",")
66
67         # Write data JSON
68         $Stream.WriteLine("`t""data"": [")
69         $Stream.Flush()
70
71         $chunksize = 250
72         $chunkarray = @()
73         $parts = [math]::Ceiling($coll.Count / $chunksize)
74
75         Write-Info "Chunking output in $chunksize item sections"
76         for($n=0; $n -lt $parts; $n++){
77             $start = $n * $chunksize
78             $end = (($n+1)*$chunksize)-1
79             $chunkarray += ,@($coll[$start..$end])
80         }
81         $Count = $chunkarray.Count
82
83         $chunkcounter = 1
84         $jsonout = ""
85         ForEach ($chunk in $chunkarray) {
86             Write-Info "Writing JSON chunk $chunkcounter/$Count"
87             $jsonout = ConvertTo-Json($chunk)
88             $jsonout = $jsonout.trimstart("`r`n").trimend("`r`n")
89             $Stream.Write($jsonout)
90             If ($chunkcounter -lt $Count) {
91                 $Stream.WriteLine(",")
92             } Else {
93                 $Stream.WriteLine("")
94             }
95             $Stream.Flush()
96             $chunkcounter += 1
97         }
98         $Stream.WriteLine("`t]")
99         $Stream.WriteLine("}")
100     } finally {
101         $Stream.close()
102     }
103 }
104
105 function Invoke-AzureHound {
106     [CmdletBinding()]
107     Param(
108         [Parameter(Mandatory=$False)][String]$TenantID = $null,
109         [Parameter(Mandatory=$False)][String]$OutputDirectory = $(Get-Location),[ValidateNo
110         [Parameter(Mandatory=$False)][Switch]$Install = $null)
111
112     if ($Install){
113         Install-Module -Name Az -AllowClobber
114         Install-module -Name AzureADPreview -AllowClobber
115     }
116
117     $Modules = Get-InstalledModule
118     if ($Modules.Name -notcontains 'Az.Accounts' -and $Modules.Name -notcontains 'Azure
```







































```
1427         $null = $Coll.Add($AppRight)
1428
1429     }
1430 }
1431 New-Output -Coll $Coll -Type "applicationadmins" -Directory $OutputDirectory
1432 Write-Info "Done processing Application Admins"
1433
1434 # Cloud Application Admins - Can create new secrets for application service princip
1435 # Write to cloudappadmins.json
1436 Write-Info "Processing Cloud Application Admins"
1437 $Coll = New-Object System.Collections.ArrayList
1438 $CloudAppAdmins = $UserRoles | Where-Object {$_.RoleID -match '158c047a-c907-4556-b
1439 $SPsWithAzureAppAdminRole = $UserRoles | Where-Object {$_.RoleID -match '158c047a-c
1440 $AppsWithAppAdminRole = ForEach ($SP in $SPsWithAzureAppAdminRole) {
1441     $AppWithRole = $SPOS | ?{$_.ServicePrincipalID -Match $SP.UserID}
1442     $AppWithRole
1443 }
1444 $CloudAppAdminRights = ForEach ($Principal in $AppAdmins) {
1445
1446     $TargetApps = $AppsWithAppAdminRole
1447
1448     ForEach ($TargetApp in $TargetApps) {
1449
1450         $AppRight = [PSCustomObject]@{
1451             AppAdminID           = $Principal.UserID
1452             AppAdminType         = $Principal.UserType
1453             AppAdminOnPremID     = $Principal.UserOnPremID
1454             TargetAppID          = $TargetApp.AppID
1455         }
1456
1457         $null = $Coll.Add($AppRight)
1458
1459     }
1460
1461     ForEach ($TargetApp in $SPswithoutRoles) {
1462
1463         $AppRight = [PSCustomObject]@{
1464             AppAdminID           = $Principal.UserID
1465             AppAdminType         = $Principal.UserType
1466             AppAdminOnPremID     = $Principal.UserOnPremID
1467             TargetAppID          = $TargetApp.AppID
1468         }
1469
1470         $null = $Coll.Add($AppRight)
1471     }
1472 }
1473 New-Output -Coll $Coll -Type "cloudappadmins" -Directory $OutputDirectory
1474 Write-Info "Done processing Cloud Application Admins"
```

```
1475
1476     Write-Host "Compressing files"
1477     $location = Get-Location
1478     $name = $date + "-azurecollection"
1479     If($OutputDirectory.path -eq $location.path){
1480         $jsonpath = $OutputDirectory.Path + [IO.Path]::DirectorySeparatorChar + "$date-"
1481         $destinationpath = $OutputDirectory.Path + [IO.Path]::DirectorySeparatorChar +
1482     }
1483     else{
1484         $jsonpath = $OutputDirectory + [IO.Path]::DirectorySeparatorChar + "$date-*.json"
1485         $destinationpath = $OutputDirectory + [IO.Path]::DirectorySeparatorChar + "$name.zip"
1486     }
1487
1488     $error.Clear()
1489     try {
1490         Compress-Archive $jsonpath -DestinationPath $destinationpath
1491     }
1492     catch {
1493         Write-Host "Zip file creation failed, JSON files may still be importable."
1494     }
1495     if (!$error) {
1496         Write-Host "Zip file created: $destinationpath"
1497         rm $jsonpath
1498         Write-Host "Done! Drag and drop the zip into the BloodHound GUI to import data."
1499     }
1500 }
```