



Sign in

This repository has been archived by the owner on Sep 2, 2022. It is now read-only.

BloodHoundAD / SharpHound3

Public archive



Notifications



Fork

155



Star

520

<> Code



Issues

7



Pull requests

9



Actions



Projects



Security



Insights

SharpHound3 / SharpHound3 / LdapBuilder.cs



136 lines (120 loc) · 6.41 KB

Code

Blame

Raw



```
1  using System.Collections.Generic;
2  using System.Linq;
3  using SharpHound3.Enums;
4
5  namespace SharpHound3
6  {
7      internal class LdapBuilder
8      {
9          /// <summary>
10         /// Builds the necessary attributes and ldap query for the specified set of options
11         /// </summary>
12         /// <param name="methods"></param>
13         /// <returns></returns>
14         internal static LdapQueryData BuildLdapQuery(CollectionMethodResolved methods)
15         {
16             var ldapFilterParts = new List<string>();
17             var ldapProperties = new List<string>();
18
19             //We always want these properties to ensure we can at least pass type finding: "samaccountname"
20             ldapProperties.AddRange(Helpers.ResolutionProps);
21             ldapProperties.Add("samaccountname");
22             //Add this property to check for GMSAs
23             ldapProperties.Add("msds-groupmsamembership");
24             //LAPS is weird and several collection methods depend on it, but its easier to just hav
```

```
25         ldapProperties.Add("ms-mcs-admpwdexpirationtime");
26
27         //Add the operatingsystem property for WindowsOnly so we can pre-filter hosts
28         if (Options.Instance.WindowsOnly)
29             ldapProperties.Add("operatingsystem");
30
31         //Group membership collection
32         if (methods.HasFlag(CollectionMethodResolved.Group))
33         {
34             ldapFilterParts.Add("(|(samaccounttype=268435456)(samaccounttype=268435457)(samaccounttype=268435458)");
35             ldapProperties.AddRange(new[] { "member", "primarygroupid" });
36         }
37
38         //Computer collection methods: ask for non-disabled computer objects
39         if (methods.HasFlag(CollectionMethodResolved.LocalAdmin) ||
40             methods.HasFlag(CollectionMethodResolved.Sessions) ||
41             methods.HasFlag(CollectionMethodResolved.LoggedOn) || methods.HasFlag(CollectionMethodResolved.DCOM) ||
42             methods.HasFlag(CollectionMethodResolved.DCOM) || methods.HasFlag(CollectionMethodResolved.DCOM))
43         {
44             ldapFilterParts.Add("&(sAMAccountType=805306369)(!(UserAccountControl:1.2.840.113556.1.4.1.1502:=1))");
45         }
46
47         //ACL Collection
48         if (methods.HasFlag(CollectionMethodResolved.ACL))
49         {
50             ldapFilterParts.Add("(|(samAccountType=805306368)(samAccountType=805306369)(samAccountType=805306370)");
51             ldapProperties.AddRange(new[]
52             {
53                 "ntsecuritydescriptor", "displayname", "name"
54             });
55         }
56
57         //Trust enumeration
58         if (methods.HasFlag(CollectionMethodResolved.Trusts))
59         {
60             ldapFilterParts.Add("(objectclass=domain)");
61         }
62
63         //Object Properties
64         if (methods.HasFlag(CollectionMethodResolved.ObjectProps))
65         {
66             ldapFilterParts.Add("(|(samaccounttype=268435456)(samaccounttype=268435457)(samaccounttype=268435458)");
67             ldapProperties.AddRange(new[]
68             {
69                 "pwdlastset", "lastlogon", "lastlogontimestamp",
70                 "sidhistory", "useraccountcontrol", "operatingsystem",
```

```
71         "operatingsystemservicepack", "serviceprincipalname", "displayname", "mail", "t
72         "homedirectory", "description", "admincount", "userpassword", "gpcfilesyspath", "obj
73         "msds-behavior-version", "objectguid", "name", "gpoptions", "msds-allowedToDeleg
74         "sidhistory"
75     });
76 }
77
78 //Container enumeration
79 if (methods.HasFlag(CollectionMethodResolved.Container))
80 {
81     ldapFilterParts.Add("(&(&(objectcategory=groupPolicyContainer)(flags=*))&(name=*))");
82     ldapProperties.AddRange(new[] { "gplink", "gpoptions", "name", "displayname" });
83 }
84
85 //GPO Local group enumeration
86 if (methods.HasFlag(CollectionMethodResolved.GPOLocalGroup))
87 {
88     //ldapFilterParts.Add("(&(&(objectcategory=groupPolicyContainer)(flags=*))&(name=*))");
89     //ldapProperties.AddRange(new[] { "gpcfilesyspath", "displayname" });
90     ldapFilterParts.Add("(&(&(objectcategory=organizationalUnit)(objectclass=domain))&(name=*))");
91     ldapProperties.AddRange(new[] { "gplink", "name" });
92 }
93
94 //SPN Target Enumeration
95 if (methods.HasFlag(CollectionMethodResolved.SPNTargets))
96 {
97     ldapFilterParts.Add("(&(samaccounttype=805306368)(serviceprincipalname=*))");
98     ldapProperties.AddRange(new[]
99     {
100         "serviceprincipalname"
101     });
102 }
103
104 //Take our query parts, and join them together
105 var finalFilter = string.Join(" ", ldapFilterParts.ToArray());
106 //Surround the filters with (|), which will OR them together
107 finalFilter = ldapFilterParts.Count == 1 ? ldapFilterParts[0] : $"(|{finalFilter})";
108
109 //Add the user specified filter if it exists
110 var userFilter = Options.Instance.LdapFilter;
111 if (userFilter != null)
112 {
113     finalFilter = $"(&({finalFilter})({userFilter}))";
114 }
115
116 if (Options.Instance.CollectAllProperties)
```

```
117         {
118             ldapProperties = new List<string>();
119             ldapProperties.Add("*");
120         }
121
122         //Distinct the attributes
123         return new LdapQueryData
124         {
125             LdapFilter = finalFilter,
126             LdapProperties = ldapProperties.Distinct().ToArray()
127         };
128     }
129 }
130
131 internal class LdapQueryData
132 {
133     public string LdapFilter { get; set; }
134     public string[] LdapProperties { get; set; }
135 }
136 }
```