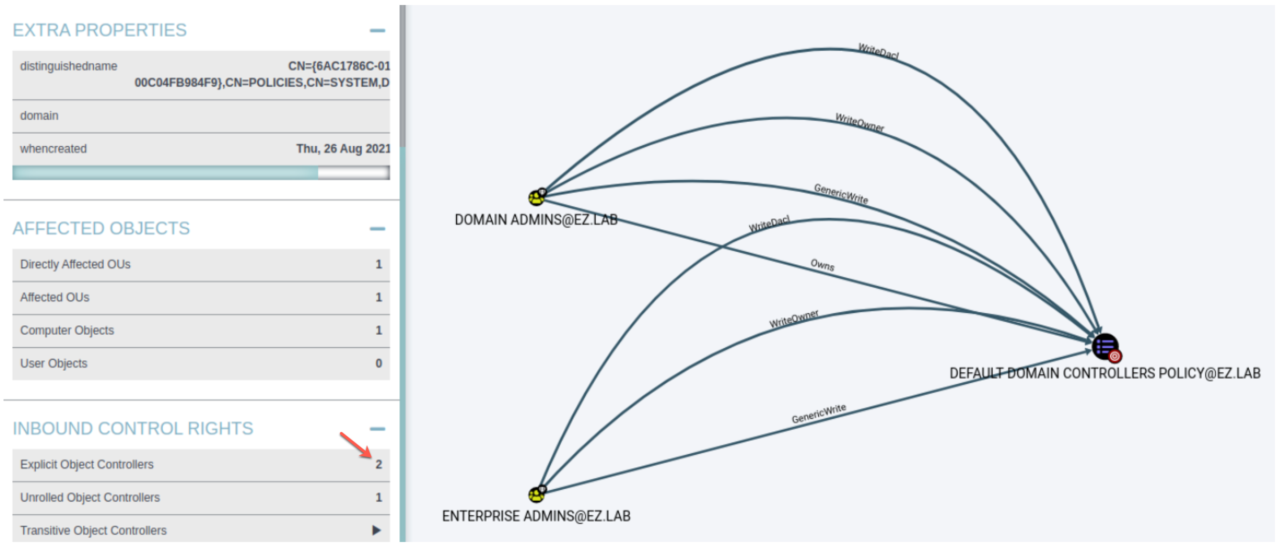
Granularize Your Active Directory Reconnaissance Game Part 2

fortalicesolutions.com/posts/granularize-your-active-directory-reconnaissance-game-part-2



EXTRA PROPERTIES

distinguishedname	CN={6AC1786C-01 00C04FB984F9},CN=POLICIES,CN=SYSTEM,D
domain	
whencreated	Thu, 26 Aug 2021

AFFECTED OBJECTS

Directly Affected OUs	1
Affected OUs	1
Computer Objects	1
User Objects	0

INBOUND CONTROL RIGHTS

Explicit Object Controllers	2
Unrolled Object Controllers	1
Transitive Object Controllers	▶

Granularize Your Active Directory Reconnaissance Game Part 2

June 15, 2022

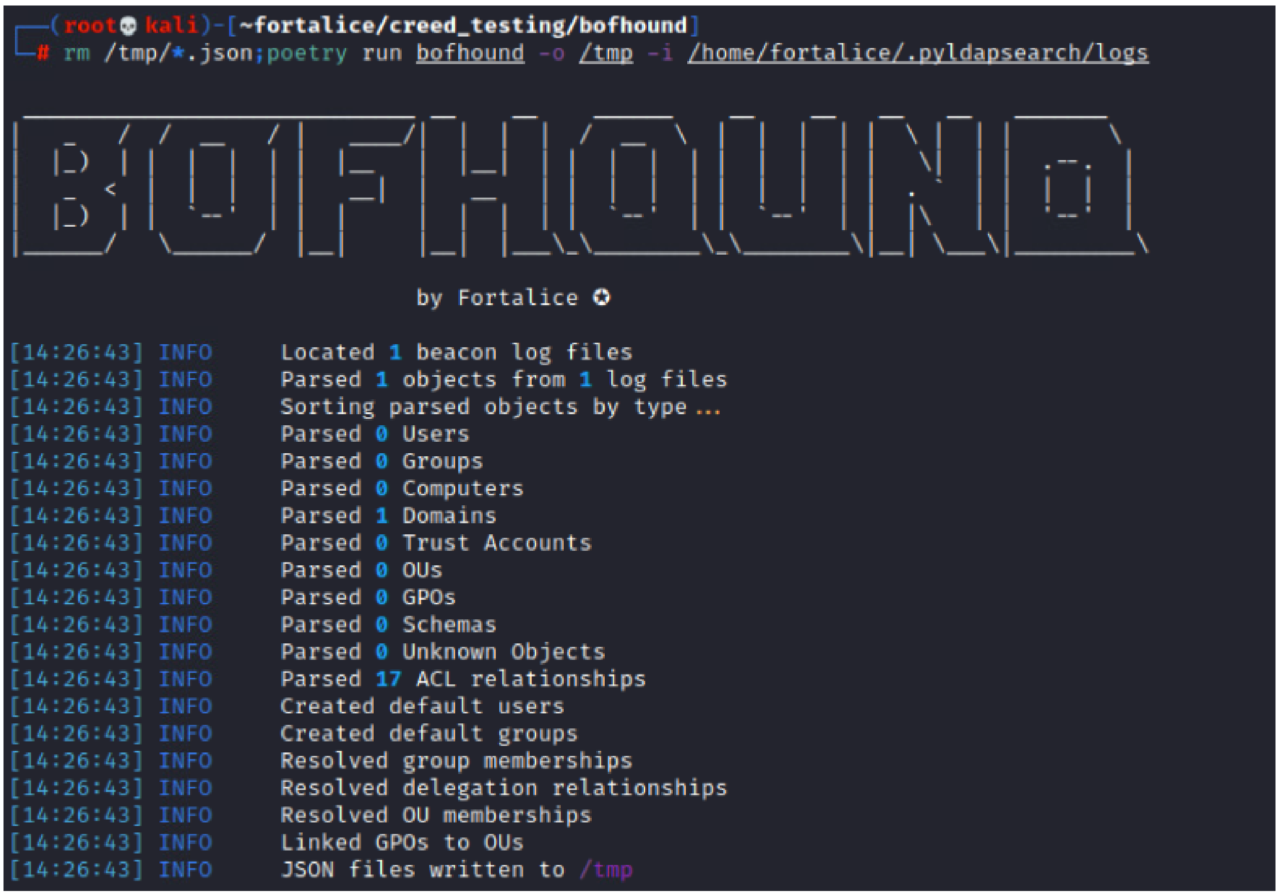
Matthew Creel

Last month Fortalice open-sourced BOFHound, an offline BloodHound ingestor for raw ldapsearch results. Along with BOFHound, we released a companion tool for it, pyldapsearch, and submitted a pull request to TrustedSec's CS-Situational-Awareness-BOF modifying the ldapsearch BOF to include the nTSecurityDescriptor attribute. Adam Brown (@coffeegist) wrote a post accompanying the release, which covered much of the tool's background, including blue team strategies for detecting BloodHound useful and the red team's reversion to more manual LDAP querying techniques. This blog will serve as a follow-up to that post, covering some usage strategies for ldapsearch + BOFHound and going into the updates that were recently pushed to BOFHound in version 0.1.0.

ACL Mapping

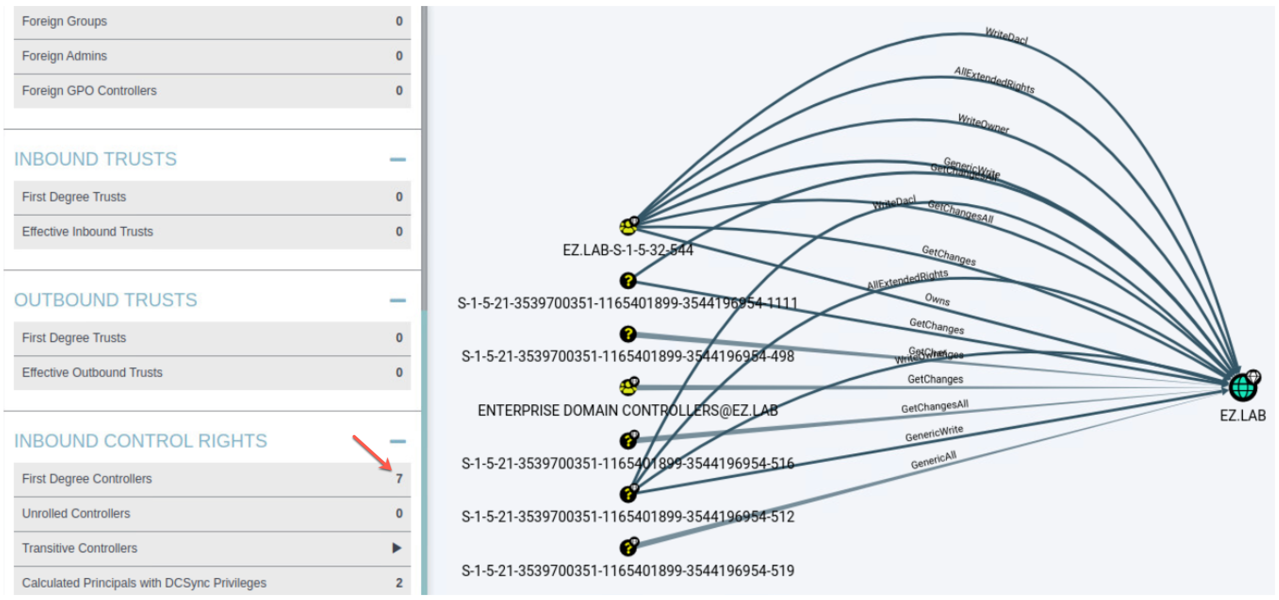
The initial release of BOFHound aimed to aid in visualizing manually queried LDAP data by parsing those results so that group memberships (`MemberOf` edges) and ACL relationships (edges such as `GenericAll`, `Owns`, `ForceChangePassword`) could be imported into BloodHound.

Let's say your goal is to simply find abuse avenues over your target domain - you can start by issuing a simple LDAP query (using pyldapsearch or the ldapsearch BOF) with the filter `'(objectClass=domain)'`. Next, we'll run BOFHound over the log file and parse that single domain object:



```
┌──(root💀 kali)-[~fortalice/creed_testing/bofhound]
└─# rm /tmp/*.json;poetry run bofhound -o /tmp -i /home/fortalice/.pyldapsearch/logs

                              by Fortalice ✪

[14:26:43] INFO     Located 1 beacon log files
[14:26:43] INFO     Parsed 1 objects from 1 log files
[14:26:43] INFO     Sorting parsed objects by type...
[14:26:43] INFO     Parsed 0 Users
[14:26:43] INFO     Parsed 0 Groups
[14:26:43] INFO     Parsed 0 Computers
[14:26:43] INFO     Parsed 1 Domains
[14:26:43] INFO     Parsed 0 Trust Accounts
[14:26:43] INFO     Parsed 0 OUs
[14:26:43] INFO     Parsed 0 GPOs
[14:26:43] INFO     Parsed 0 Schemas
[14:26:43] INFO     Parsed 0 Unknown Objects
[14:26:43] INFO     Parsed 17 ACL relationships
[14:26:43] INFO     Created default users
[14:26:43] INFO     Created default groups
[14:26:43] INFO     Resolved group memberships
[14:26:43] INFO     Resolved delegation relationships
[14:26:43] INFO     Resolved OU memberships
[14:26:43] INFO     Linked GPOs to OUs
[14:26:43] INFO     JSON files written to /tmp
```

Several JSON files will be written, which we'll upload to BloodHound. Locating the domain object in the BloodHound GUI, we can head to the *Inbound Control Rights* section and click the number associated with *First Degree Controllers*. This will us a graph that looks something like this, where we can see objects (ones we haven't queried) by their SIDs and their rights over the domain object.


EZ.LAB-S-1-5-32-544
S-1-5-21-3539700351-1165401899-3544196954-1111
S-1-5-21-3539700351-1165401899-3544196954-498
ENTERPRISE DOMAIN CONTROLLERS@EZ.LAB
S-1-5-21-3539700351-1165401899-3544196954-516
S-1-5-21-3539700351-1165401899-3544196954-512
S-1-5-21-3539700351-1165401899-3544196954-519
EZ.LAB
WriteDacl
AllExtendedRights
WriteOwner
GenericWrite
GetChangesAll
WriteDacl
GetChangesAll
GetChanges
AllExtendedRights
Owns
GetChanges
WriteOwner
GetChanges
GetChangesAll
GenericWrite
GenericAll

Foreign Groups	0
Foreign Admins	0
Foreign GPO Controllers	0

INBOUND TRUSTS

First Degree Trusts	0
Effective Inbound Trusts	0

OUTBOUND TRUSTS

First Degree Trusts	0
Effective Outbound Trusts	0

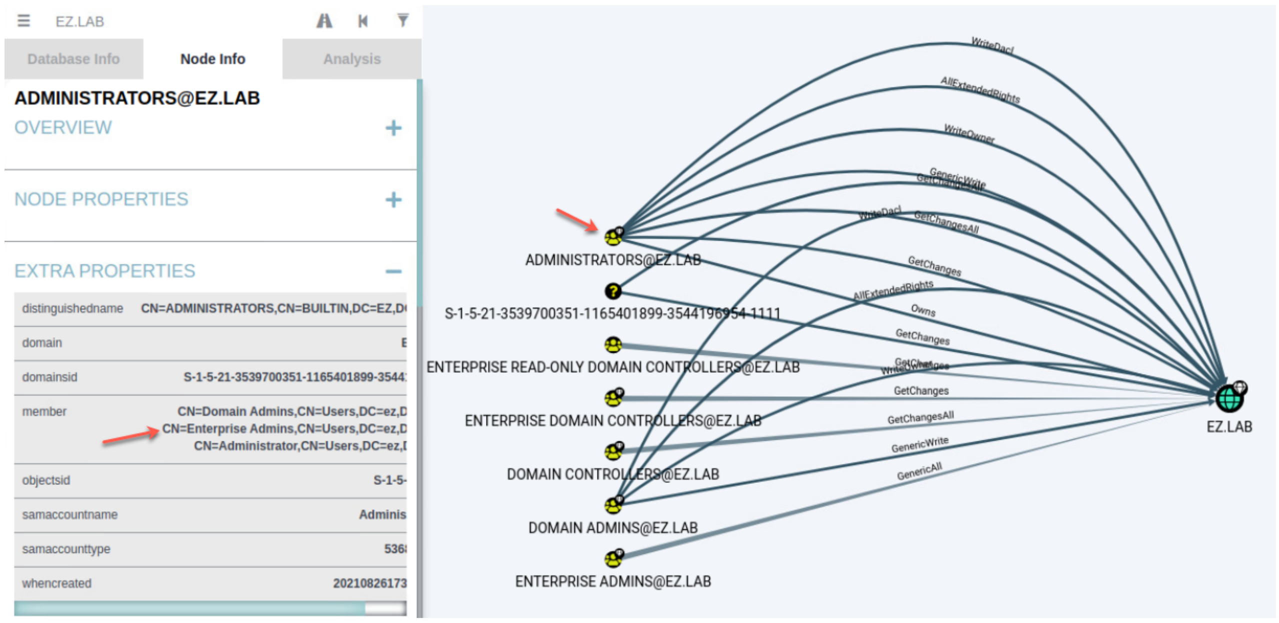
INBOUND CONTROL RIGHTS

First Degree Controllers	7
Unrolled Controllers	0
Transitive Controllers	▶
Calculated Principals with DCSync Privileges	2

With this as a starting point towards our goal, we've got more objects to query. Looking at those SIDs, I know `S-1-5-32-544` is the Administrators group, and the SIDs ending in 512, 516, and 519 are Domain Controllers, Domain Admins, and Enterprise Admins. I'm not sure what the SID ending in 498 is off the top of my head and I also know the SID ending in 1111 belongs to an object which no longer exists in this domain. Accordingly, I'll run some more LDAP queries to gather the objects in our result set:

```
'(sAMAccountName=Administrators)'
'(sAMAccountName=Domain Admins)'
'(sAMAccountName=Enterprise Admins)'
'(sAMAccountName=Domain Controllers)'
'(objectSid=S-1-5-21-3539700351-1165401899-3544196954-498)'
```

Rerun BOFHound, load the output into BloodHound and now the graph is looking more filled in. Additionally, if we now click on one of the group objects we can see the distinguished names of members:



ADMINISTRATORS@EZ.LAB

distinguishedname	CN=ADMINISTRATORS,CN=BUILTIN,DC=EZ,D
domain	E
domainsid	S-1-5-21-3539700351-1165401899-3544

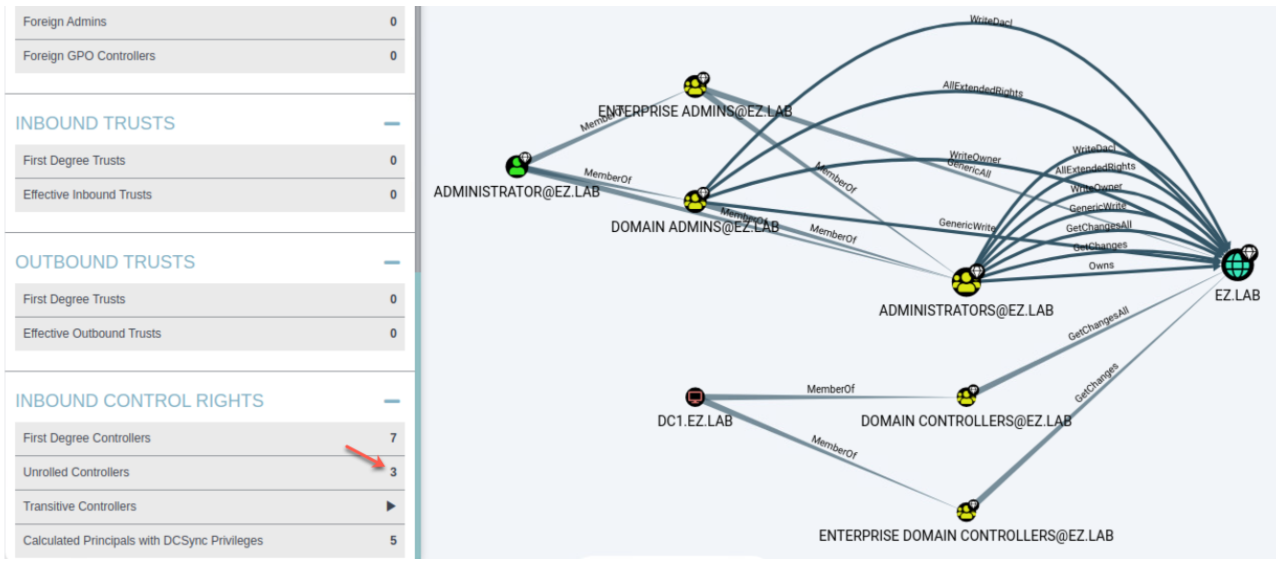
member	CN=Domain Admins,CN=Users,DC=ez,D
	CN=Enterprise Admins,CN=Users,DC=ez,D
	CN=Administrator,CN=Users,DC=ez,

objectsid	S-1-5-
samaccountname	Adminis
samaccounttype	536
whencreated	20210826173

We can use this information to spider out further and start revealing *Unrolled Controllers*, or objects with nested group memberships granting nth-degree object control over the domain. My example domain isn't heavily populated so member objects of these groups are minimal. In client environments, it'd likely be tedious to query each manually by distinguished name, but the good news is you have options. Option 1 is to query with a filter on the `memberOf` field, like this: `'(memberOf=CN=Administrators,CN=Builtin,DC=ez,DC=lab)'`. This will return all *direct* members of the Administrators group. Option 2 is to query all *nested* members of the target group by using a specific LDAPOID, like this: `'(memberOf:1.2.840.113556.1.4.1941:CN=Administrators,CN=Builtin,DC=ez,DC=lab)'`. Something to consider with both of those options is that the `memberOf` attribute is not indexed by Active Directory, meaning the query is likely to visit a high number of LDAP objects, even if few are ultimately relevant. This is one of the detection strategies mentioned in the original BOFHound post. In environments where this is a detection consideration, I'll sometimes just use a bash loop or `xargs` to individually query the distinguished names with a small sleep between each query (if using the ldapsearch BOF, this probably requires some aggressor scripting). However, in my small lab I'll just query the members of these groups by sAMAccountName:

```
'(sAMAccountName=Administrator)'
'(sAMAccountName=DC1$)'
```

Rerun BOFHound, reimport to BloodHound and we can now see the unrolled controllers related to the target domain:



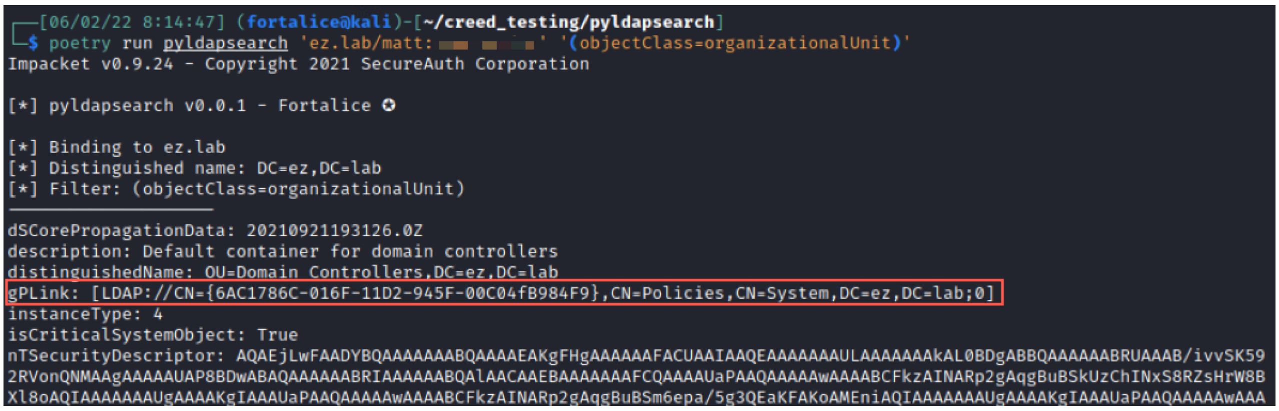
Foreign Admins	0
Foreign GPO Controllers	0
INBOUND TRUSTS	
First Degree Trusts	0
Effective Inbound Trusts	0
OUTBOUND TRUSTS	
First Degree Trusts	0
Effective Outbound Trusts	0
INBOUND CONTROL RIGHTS	
First Degree Controllers	7
Unrolled Controllers	3
Transitive Controllers	▶
Calculated Principals with DCSync Privileges	5

At this point, you can rinse and repeat what we've done to keep enumerating layers of attack paths. We could start exploring any unknown objects that pop up with the *Transitive Controllers* query on the domain, but we've also parsed ACLs for several users and groups which we could start to hunt paths to individually.

In practice, the approach demonstrated here - one or two LDAP queries followed by BOFHound and importing - is fairly slow, so you may choose to run larger sets of queries or more aggressive queries before each BOFHound pass. However, the approach here does demonstrate that as the operator, you are in total control over how granular you want AD reconnaissance to be, all while maintaining the ability to utilize BloodHound.

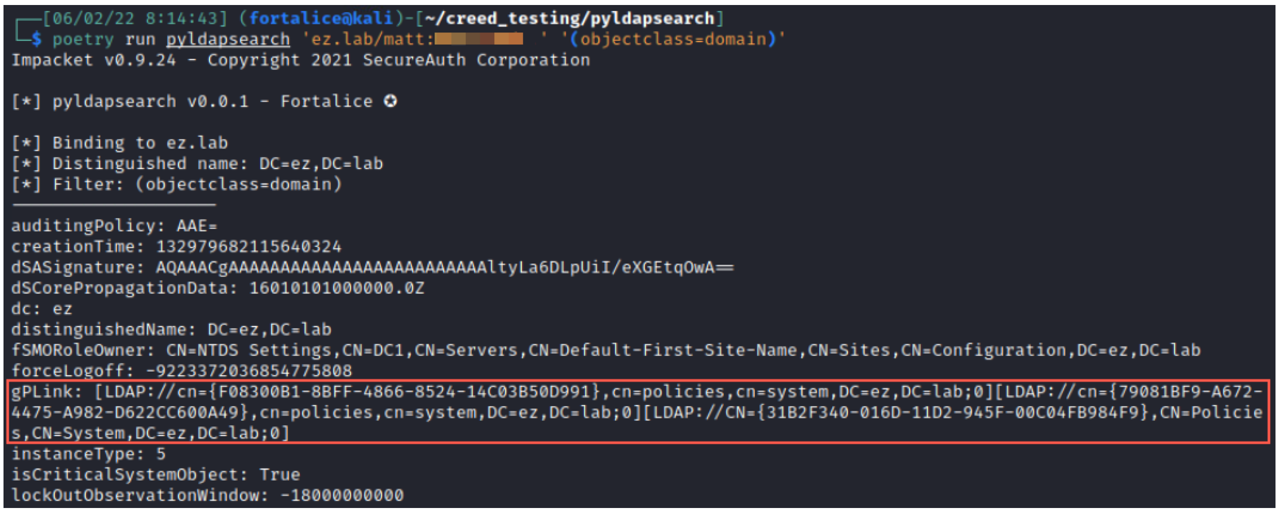
GPO and OU Mapping

All of that was possible in the original release so what's new in BOFHound 0.1.0? In this release, we've added support for OU and GPO objects, as well as parsing out domain trust relationships (those can be queried with `'(objectClass=trustedDomain)'`). Targeted enumeration of OUs and linked GPOs will work similarly to the groups and group members from above. As an example, in my lab, I'll query all OUs, but if you're targeting some specific subset of machines (i.e., Citrix servers), you could try a query filter similar to `'(&(objectClass=organizationalUnit)(|(ou=*citrix*)(ou=*ctx*)))'`.



```
┌──[06/02/22 8:14:47] (fortalice㉿kali)-[~/creed_testing/pyldapsearch]
└─$ poetry run pyldapsearch 'ez.lab/matt:[redacted]' '(objectClass=organizationalUnit)'
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] pyldapsearch v0.0.1 - Fortalice ✪

[*] Binding to ez.lab
[*] Distinguished name: DC=ez,DC=lab
[*] Filter: (objectClass=organizationalUnit)
────────────────────

dSCorePropagationData: 20210921193126.0Z
description: Default container for domain controllers
distinguishedName: OU=Domain Controllers,DC=ez,DC=lab
gPLink: [LDAP://CN={6AC1786C-016F-11D2-945F-00C04fB984F9},CN=Policies,CN=System,DC=ez,DC=lab;0]
instanceType: 4
isCriticalSystemObject: True
nTSecurityDescriptor: AQAEjLwFAADYBQAAAAAAABQAAAAEAKgFHgAAAAAAFACUAAIAAQEAAAAAAAAULAAAAAAAkAL0BDgABBQAAAAAABRUAAAB/ivvSK59
2RVonQNMAAgAAAAAUAP8BDwABAQAAAAAABRIAAAAAABQAlAACAAEBAAAAAAAFCQAAAAUaPAAQAAAAAwAAAABCFkzAINARp2gAqgBuBSkUzChINxS8RZsHrW8B
Xl8oAQIAAAAAAAUgAAAAKgIAAAUaPAAQAAAAAwAAAABCFkzAINARp2gAqgBuBSm6epa/5g3QEaKFAKoAMEniAQIAAAAAAAUgAAAAKgIAAAUaPAAQAAAAAwAAA
```

The *gpLink* attribute contains the distinguished names of associated group policy objects. GPOs can also be linked directory to domain objects:

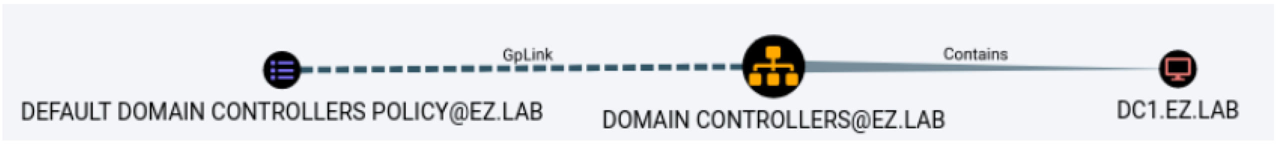


```
┌──[06/02/22 8:14:43] (fortalice㉿kali)-[~/creed_testing/pyldapsearch]
└─$ poetry run pyldapsearch 'ez.lab/matt:[redacted]' '(objectclass=domain)'
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] pyldapsearch v0.0.1 - Fortalice ✪

[*] Binding to ez.lab
[*] Distinguished name: DC=ez,DC=lab
[*] Filter: (objectclass=domain)
────────────────────

auditingPolicy: AAE=
creationTime: 132979682115640324
dSASignature: AQAAACgAAAAAAAAAAAAAAAAAAAAAAAAltyLa6DLpUiI/eXGEtqOwA==
dSCorePropagationData: 16010101000000.0Z
dc: ez
distinguishedName: DC=ez,DC=lab
fSMORoleOwner: CN=NTDS Settings,CN=DC1,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=ez,DC=lab
forceLogoff: -9223372036854775808
gPLink: [LDAP://cn={F08300B1-8BFF-4866-8524-14C03B50D991},cn=policies,cn=system,DC=ez,DC=lab;0][LDAP://cn={79081BF9-A672-
4475-A982-D622CC600A49},cn=policies,cn=system,DC=ez,DC=lab;0][LDAP://CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policie
s,CN=System,DC=ez,DC=lab;0]
instanceType: 5
isCriticalSystemObject: True
lockOutObservationWindow: -18000000000
```

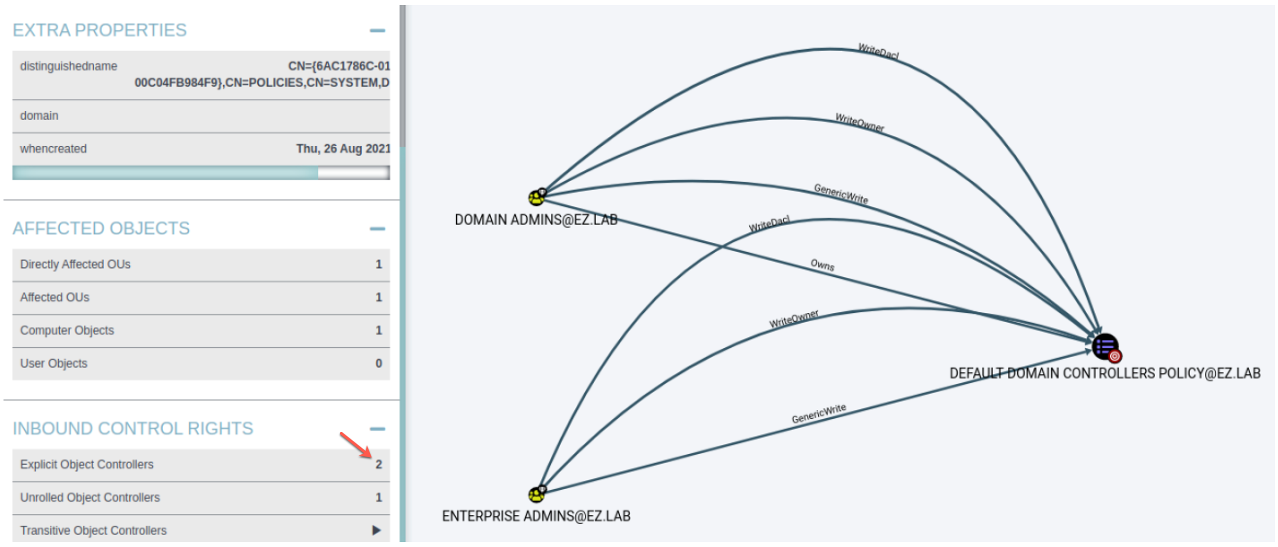
Now we query for the GPOs linked to the specific OUs we're interested in:

```
'(cn={F08300B1-8BFF-4866-8524-14C03B50D991})'
'(cn={6AC1786C-016F-1102-945F-00C04fB984F9})'
...
```

Run BOFHound to collect the data from our logs, upload to BloodHound and we can see how GPOs filter down to OUs and other objects.


GpLink
Contains
DEFAULT DOMAIN CONTROLLERS POLICY@EZ.LAB
DOMAIN CONTROLLERS@EZ.LAB
DC1.EZ.LAB

If we're hunting for modifiable GPOs, we can also view the inbound controllers and start down the ACL path again:



EXTRA PROPERTIES

distinguishedname	CN={6AC1786C-01 00C04FB984F9},CN=POLICIES,CN=SYSTEM,D
domain	
whencreated	Thu, 26 Aug 2021

AFFECTED OBJECTS

Directly Affected OUs	1
Affected OUs	1
Computer Objects	1
User Objects	0

INBOUND CONTROL RIGHTS

Explicit Object Controllers	2
Unrolled Object Controllers	1
Transitive Object Controllers	▶

Wrapping Up

The majority of the LDAP query filters used as examples throughout this post are very simple - mostly consisting of a sole condition to return a single object of interest. This is sufficient for populating targeted ACL relationships, however, if you're looking for slightly more wide-ranging queries to help populate your BloodHound database, you'll want to branch out from this structure. Two great resources to help you get started with manual LDAP searching can be found here by Hope Walker (@Icemoonhsv) and here by Rob Fuller (@mubix).

For additional information on Fortalice Solutions' service offerings, contact the team via email at watchmen@fortalicesolutions.com.

Offensive Cybersecurity Operations