Task 01 : Vulnerability Assessment & Penetration Testing Report

1. Summary

A vulnerability assessment was conducted on the Metasploitable 2 system using open-source security tools. The objective was to identify security weaknesses, assess their risk levels, and recommend mitigations.

The assessment revealed several high-severity vulnerabilities, including exposed database and remote management services, outdated web components, and weak network configurations. These issues could allow an attacker to gain unauthorized access, exploit services, or exfiltrate sensitive information.

Overall Risk Level: High
Total Vulnerabilities Identified: 60+

- High Severity: 22
- Medium Severity: 38
- Low Severity: 6

Immediate remediation should focus on closing unnecessary ports, restricting service exposure, and applying software updates.

2. Scope and Methodology

Scope

- **Target:** Metasploitable 2 virtual machine
- **IP Range:** 192.168.56.101 (local isolated lab)
- **Goal:** Identify, analyze, and report vulnerabilities using open-source tools.
- **Tools Used:**
    - *Nmap* – Port and service enumeration
    - *OpenVAS* – Automated vulnerability scanning (CVE-based)
    - *Nikto* – Web server misconfiguration detection
    - *Curl* – Header inspection and web response validation
    - *Gobuster* – Directory enumeration on web applications

Methodology (VAPT Phases)

1. **Planning:** Defined testing scope and lab setup (Kali + Metasploitable2).
2. **Discovery:** Used Nmap to identify live hosts, open ports, and running services.
3. **Scanning:** Performed vulnerability scans using OpenVAS and Nikto.

4. **Analysis:** Validated findings manually using curl and gobuster.
5. **Risk Assessment:** Assigned severity using CVSS scores and a 3×3 risk matrix.
6. **Reporting:** Compiled all results, screenshots, and mitigation steps.

3. Technical Findings

Sample

ID	Host	Port	Service	Severity	Description	Recommendation
V-001	192.168.56.101	5432	PostgreSQL	**High**	Database service exposed to all networks. Could allow remote authentication brute force or exploitation of known CVEs.	Restrict PostgreSQL to localhost or internal IPs; apply authentication and firewall rules.
V-002	192.168.56.101	1099	Java RMI	**High**	RMI registry is open, allowing remote class loading and possible RCE.	Disable RMI service or restrict to trusted IPs; apply java.rmi.server.hostname and security manager controls.
V-003	192.168.56.101	21	FTP	**High**	Anonymous FTP login enabled, allowing public file access.	Disable anonymous FTP and enforce user authentication.

For Full list please verify the sheet attached:
https://docs.google.com/spreadsheets/d/1WTzf0a3z0y5yz7gniLRU8_5zRCZq5qt5SwOhqFOZB6g/edit?usp=sharing

4. Risk Assessment

CVSS Integration:
OpenVAS generated CVSS scores were used to calculate overall risk. Each vulnerability was also rated using a qualitative Likelihood vs Impact matrix:

Likelihood	Impact	Risk
High	High	🔴 Critical
Medium	High	🟠 High

Medium	Medium	🟡 Medium
Low	Medium	🟢 Low

High-Risk Items (Priority Fixes):

- Exposed database services (PostgreSQL, MySQL)
- Remote login protocols (Telnet, FTP)
- Outdated web server (Apache)
- Java RMI & VNC services exposed

5. Recommendations Summary

Priority	Action Item	Responsible	Status
High	Disable Telnet & anonymous FTP	System Admin	Pending
High	Patch and secure Apache web server	Web Admin	Pending
High	Restrict DB ports (5432, 3306) via firewall	Network Admin	Pending
Medium	Harden SSH & SMB configurations	System Admin	Pending
Low	Limit ICMP echo responses	Network Admin	Pending

6. Conclusion

The assessment successfully identified multiple network and web-layer vulnerabilities in the Metasploitable 2 environment. These weaknesses demonstrate the importance of service hardening, regular patching, and restricted exposure of critical services.

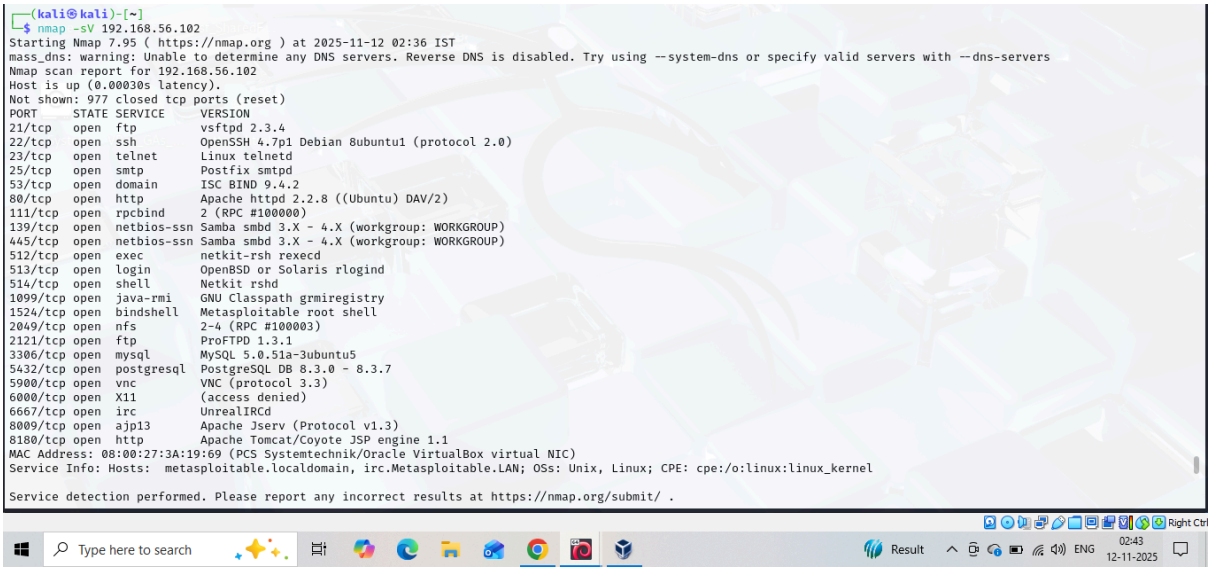
Applying the recommended remediations will significantly reduce the system's attack surface and improve its overall security posture.

It is recommended to re-scan the system after patches are applied to verify that vulnerabilities are mitigated.

7. Tools & Commands Used(With Proof's):

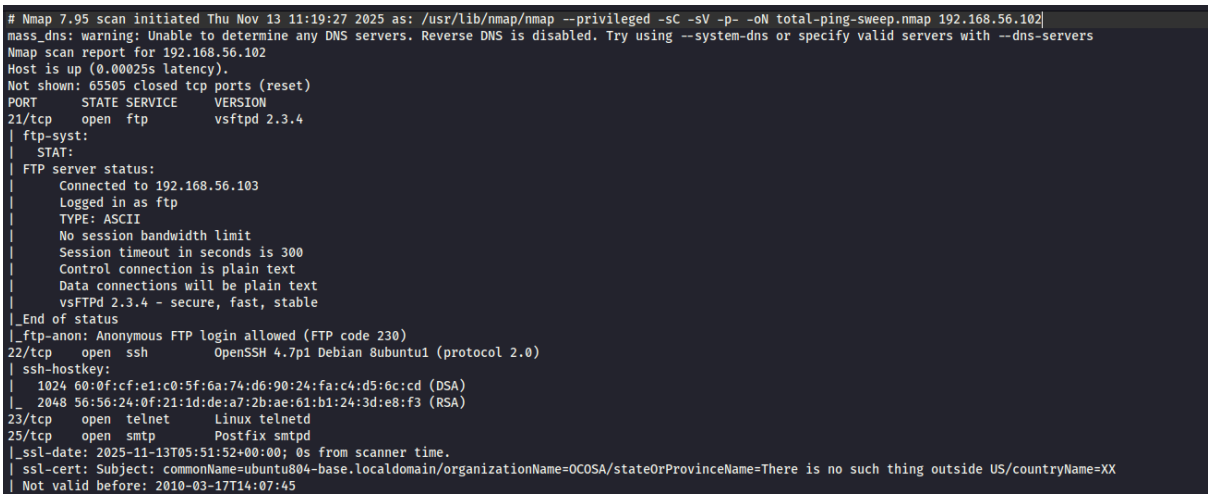
Basic Discovery:

- Find the IP address of your victim VM (Metasploitable 2).
- Run a foundational scan: nmap -sV [Metasploitable_IP] (The -sV option attempts to determine service versions)



Network discovery/Full port & service discovery

nmap -sC -sV -p- -oN total-ping-sweep.nmap 192.168.56.102



Web Application Checks/Web scanning

nikto -h http://192.168.56.102:80 -o nikto_metas.txt

```
- Nikto v2.5.0/
+ Target Host: 192.168.56.102
+ Target Port: 80
+ GET /: Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10.
+ GET /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options:
+ GET /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See:
https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/:
+ GET /index: Uncommon header 'tcn' found, with contents: list.
+ GET /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were
found: index.php. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275:
+ HEAD Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ JMGYSQUX /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ TRACE /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing:
+ GET /phpinfo.php: Output from the phpinfo() function was found.
+ GET /doc/: Directory indexing found.
+ GET /doc/: The /doc/ directory is browsable. This may be /usr/doc. See: CVE-1999-0678:
+ GET /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184:
+ GET /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184:
+ GET /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184:
+ GET /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184:
+ GET /phpMyAdmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ GET /phpMyAdmin/ChangeLog: Server may leak inodes via ETags, header found with file /phpMyAdmin/ChangeLog, inode: 92462, size: 40540, mtime: Tue Dec  9 22:54:00
2008. See: CVE-2003-1418:
+ GET /phpMyAdmin/ChangeLog: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ GET /test/: Directory indexing found.
+ GET /test/: This might be interesting.
+ GET /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information. See: CWE-552:
+ GET /icons/: Directory indexing found.
```

directory brute (discovery)

gobuster dir -u http://192.168.56.102/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -o gobuster_metas.txt

```
/index                (Status: 200) [Size: 891]
/test                 (Status: 301) [Size: 320] [--> http://192.168.56.102/test/]
/twiki                (Status: 301) [Size: 321] [--> http://192.168.56.102/twiki/]
/tikiwiki             (Status: 301) [Size: 324] [--> http://192.168.56.102/tikiwiki/]
/phpinfo              (Status: 200) [Size: 48008]
/server-status        (Status: 403) [Size: 300]
/phpMyAdmin           (Status: 301) [Size: 326] [--> http://192.168.56.102/phpMyAdmin/]
```

quick header check (non-intrusive)

curl -I http://192.168.56.102/

```
HTTP/1.1 200 OK
Date: Thu, 13 Nov 2025 18:29:47 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Content-Type: text/html
```

#OpenVas/GreenBone

1. Open the Greenbone Security Assistant (GSA) / GVM web UI and log in.
2. Go to Scans → Tasks (or Scans → Reports depending on UI version).
3. Find the task you ran (it shows status: Done/Stopped/Interrupted).
4. Click the Reports icon or open the task and click Reports.
5. In the Reports list, click the report for the scan run you want.
6. Click Download (or the export icon) and select formats to download:
   PDF — for attaching directly to the report (human readable).
   XML / GVM-XML — raw structured data (appendix / importable).

CSV — for spreadsheet import and custom sorting/filtering.

Task	Severity	High	Medium	Low	Log	False Pos.
Immediate scan of IP 192.168.56.102	10.0 (High)	22	38	6	89	0