
Comprehensive Assessment Script
– Manual (Kali, Ethical Use)
KALI
Moeez Javed

Comprehensive Assessment Script Manual (Kali, Ethical Use)

Audience: Students and interns learning how to run **authorized** reconnaissance and vulnerability assessments in a controlled lab or with written permission.

Purpose: Explain *why* combining tools matters, how to prepare a safe lab workflow, and how to run (and extend) a one-shot script that orchestrates Nmap → web checks (Nikto/Dirb/WPScan) → targeted Nmap vuln scripts → a preliminary Markdown report.

Legal & Ethics

• Run **only** on systems you own or have explicit written authorization to test.
• Use the script in a lab/VPN/segmented network.
• Disable any destructive checks; avoid denial-of-service.
• Treat outputs as sensitive and store them securely.

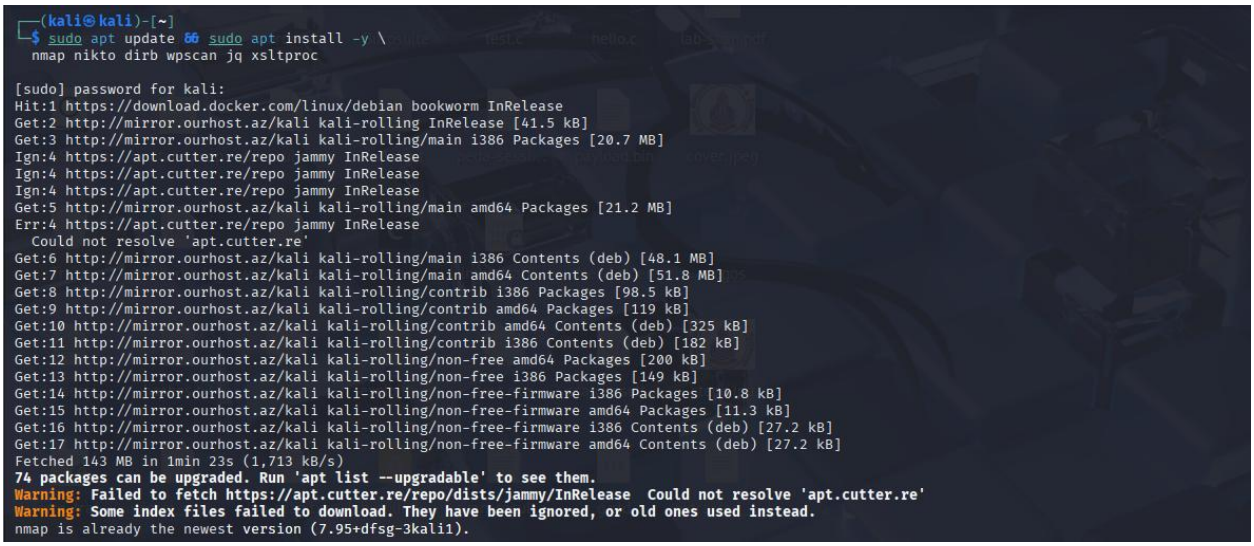
1) Why combine tools?

Security assessments are rarely a single tool job. A stitched workflow lets you: - Keep evidence together under a timestamped folder (reproducibility), - Reuse open-port lists to focus follow-up scans (efficiency), - Chain web-only actions **only** if web ports are present (signal over noise), - Produce a quick, human-readable report to brief your blue team.

2) Prerequisites (Kali/Linux)

Install/update the tools we'll call from the script.

```
sudo apt update && sudo apt install -y \
  nmap nikto dirb wpscan jq xsltproc
# optional helpers used later in the manual
```




```
sudo apt install -y curl whatweb
```

Tip: Some scans (OS detection, SYN) work best with sudo/root. Run the script with sudo inside your lab.

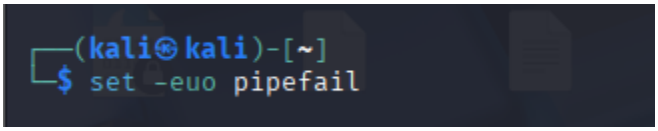
3) The orchestrator script (with safe defaults)

Create a working folder and script file:

```
mkdir -p ~/assessments && cd ~/assessments
nano assessment.sh
```

Paste the script below **exactly** (this version fixes line-break bugs and adds safety checks):


```
#!/usr/bin/env bash
set -euo pipefail
```




```
#!/bin/bash

# ================================
# Automated Security Assessment Script
# ================================

# Check if target provided
if [ -z "$1" ]; then
    echo "Usage: $0 <TARGET-IP/HOSTNAME>"
    exit 1
fi

TARGET="$1"
OUTPUT_DIR="./assessment_$(date +%Y%m%d_%H%M%S)"
mkdir -p "$OUTPUT_DIR"

echo "[*] Starting comprehensive assessment of $TARGET"
```

```
echo "[*] Results will be stored in: $OUTPUT_DIR"
echo

# ================================
# 1. Initial Nmap Scan
# ================================
echo "[*] Running initial Nmap scan (top 1000 TCP ports)..."
nmap -sS -sV -sC -O -T3 -Pn -oA "$OUTPUT_DIR/nmap_scan" "$TARGET"

# Extract open ports
PORTS=$(grep -oP '\d+/open' "$OUTPUT_DIR/nmap_scan.gnmap" | cut -d'/' -f1 | tr '\n' ',' | sed 's/,$//')

if [ -z "$PORTS" ]; then
    echo "[!] No open ports found in top 1000. Running full port scan..."
    nmap -p- -T4 -oA "$OUTPUT_DIR/nmap_full" "$TARGET"
    PORTS=$(grep -oP '\d+/open' "$OUTPUT_DIR/nmap_full.gnmap" | cut -d'/' -f1 | tr '\n' ',' | sed 's/,$//')
fi
if [ -z "$PORTS" ]; then
    echo "[!] No open ports detected on $TARGET. Exiting."
    exit 0
fi

echo "[*] Open ports detected: $PORTS"
```

```
echo
# ================================
# 2. Vulnerability Scan
# ================================
echo "[*] Running Nmap vulnerability scan on detected ports..."
nmap --script vuln -p"$PORTS" "$TARGET" -oA "$OUTPUT_DIR/nmap_vuln_scan"
echo
# ================================
# 3. Web Application Scans
# ================================
if echo "$PORTS" | grep -q "80\|443\|8080\|8443"; then
    echo "[*] Web service detected. Running Nikto, Dirb, and WPScan..."
    nikto -h "http://$TARGET" -output "$OUTPUT_DIR/nikto_scan.txt"
    dirb "http://$TARGET" -o "$OUTPUT_DIR/dirb_scan.txt"
    wpscan --url "http://$TARGET" --enumerate ap,at,cb,dbe -o "$OUTPUT_DIR/wpscan.txt"
    echo "[*] Web scans completed."
    echo
else
    echo "[*] No common web ports found. Skipping web scans."
fi
# ================================
# 4. Report Generation
# ================================
```

```
echo "[*] Generating preliminary report..."
cat > "$OUTPUT_DIR/preliminary_report.md" << EOF
# Security Assessment Report
**Target:** $TARGET
**Date:** $(date)

---

## Network Services
$(cat "$OUTPUT_DIR/nmap_scan.nmap" | grep -E "^\d+/tcp|^\d+/udp" || echo "No services detected.")
---
## Potential Vulnerabilities
$(grep -h "VULNERABLE" "$OUTPUT_DIR"/*.txt 2>/dev/null || echo "No automatic vulnerabilities detected.")
---
## Recommendations
- Review all identified services manually
- Investigate potential vulnerabilities in detail
- Perform authenticated testing where possible
EOF
echo "[*] Assessment complete!"
echo "[*] All results are saved in: $OUTPUT_DIR"
```

Save and close, then make it executable and run it:

```
chmod +x assessment.sh
sudo ./assessment.sh 10.0.2.15   # replace with your authorized lab target
```

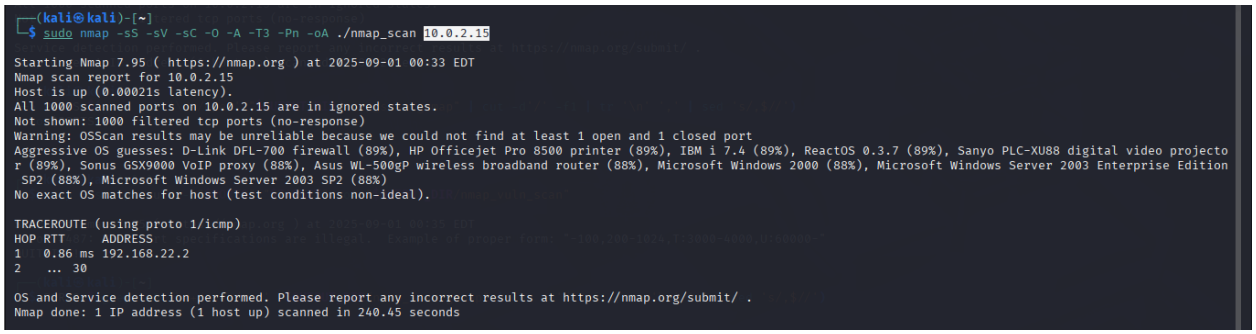
Output: a timestamped folder assessment_YYYYMMDD_HHMMSS containing Nmap/Nikto/Dirb/WPScan outputs, a run.log, and a preliminary_report.md.

4) Step-by-step: what every command does

a) Nmap core scan

```
sudo nmap -sS -sV -sC -O -A -T3 -Pn -oA ./nmap_scan 10.0.2.15
```

- -sS SYN scan (fast, less intrusive than full connect).
- -sV service/version detection.
- -sC default NSE scripts (safe).
- -O OS fingerprinting; -A adds aggressive detection (includes OS, traceroute, some scripts).
- -T3 balanced timing (safer in shared labs than -T4).
- -Pn skip host discovery (treat host as up).
- -oA write all formats (.nmap, .gnmap, .xml)



b) Extract open ports from Nmap

PORTS=$(grep -oP '\\d+/open/tcp' "$OUTPUT_DIR/nmap_scan.gnmap" | cut -d'/' -f1 | tr '\\n' ',' | sed 's/,$//')

- Greps open TCP ports from the .gnmap file, plucks just the port numbers, converts newlines to commas for reuse, and trims the trailing comma.

c) Conditional web checks

- We only run Nikto/Dirb/WPScan if web-ish ports were found (80, 443, 8080, 8443).

- nikto runs once for HTTP and once for HTTPS; failures don't stop the script (|| true).

- dirb quickly enumerates common paths; for deeper tests, switch to a bigger wordlist.

- We try a **cheap** WP fingerprint (look for wp-content / wp-includes in HTML) before launching wpscan.

d) Nmap vuln scripts

nmap --script vuln -p"$PORTS" "$TARGET" -oA "$OUTPUT_DIR/nmap_vuln_scan"

- Runs NSE scripts tagged vuln **only on discovered ports**. Treat results as *findings to be verified*, not proof of exploitation.

e) Preliminary report generation

- Builds preliminary_report.md with service lines, any "VULNERABLE" markers from tool outputs, and structured sections you can expand manually.

5) How to read the results (triage guide)

1. **Open ports & services**: Start with nmap_scan.nmap and confirm the service banner makes sense for the host's role.

2. **Vuln hints**: Review nmap_vuln_scan.nmap for CVE references **then verify manually**.

3. **Web checks**: Compare nikto_*.txt and dirb_*.txt; look for sensitive files, outdated server banners, default pages. If wpscan_*.txt exists, read the summary and plugin/theme enumeration results.

4. **Report**: Open preliminary_report.md and add human analysis: false positives, compensating controls, and prioritized actions.

6) Variations & safe extensions (optional)

- **Safer timing:** If the target is fragile, drop to -T2 and remove -A (keep -sV -sC).

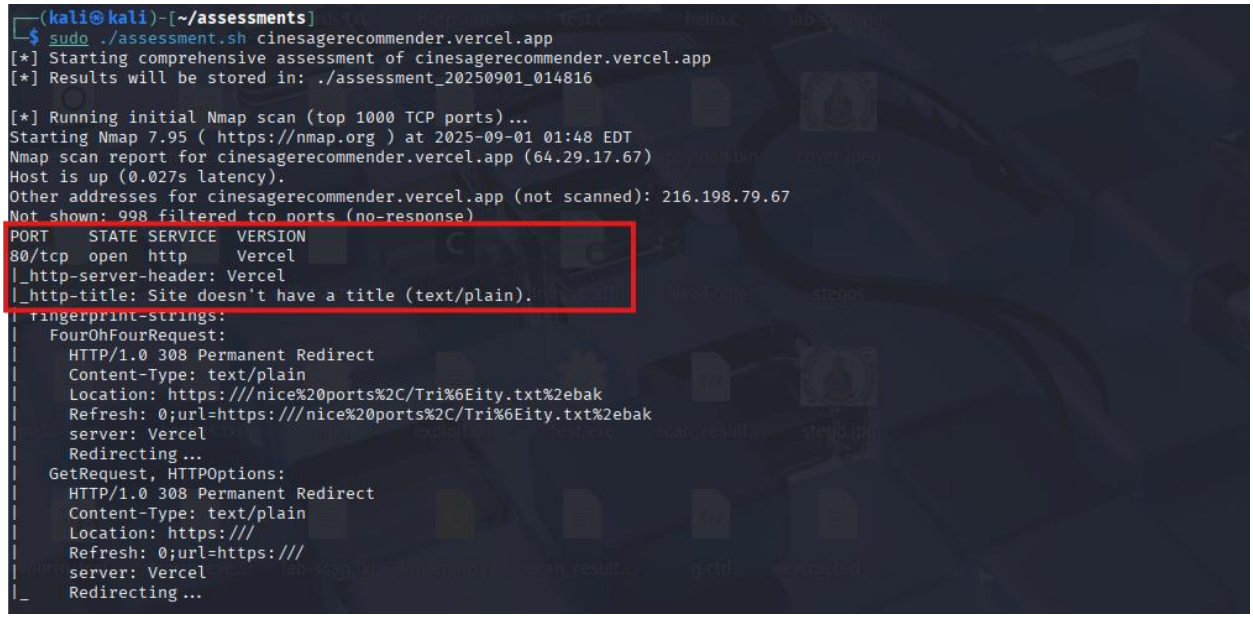
- **Scope files:** Replace single TARGET with a list and loop (while read h; do ... done < scope.txt).

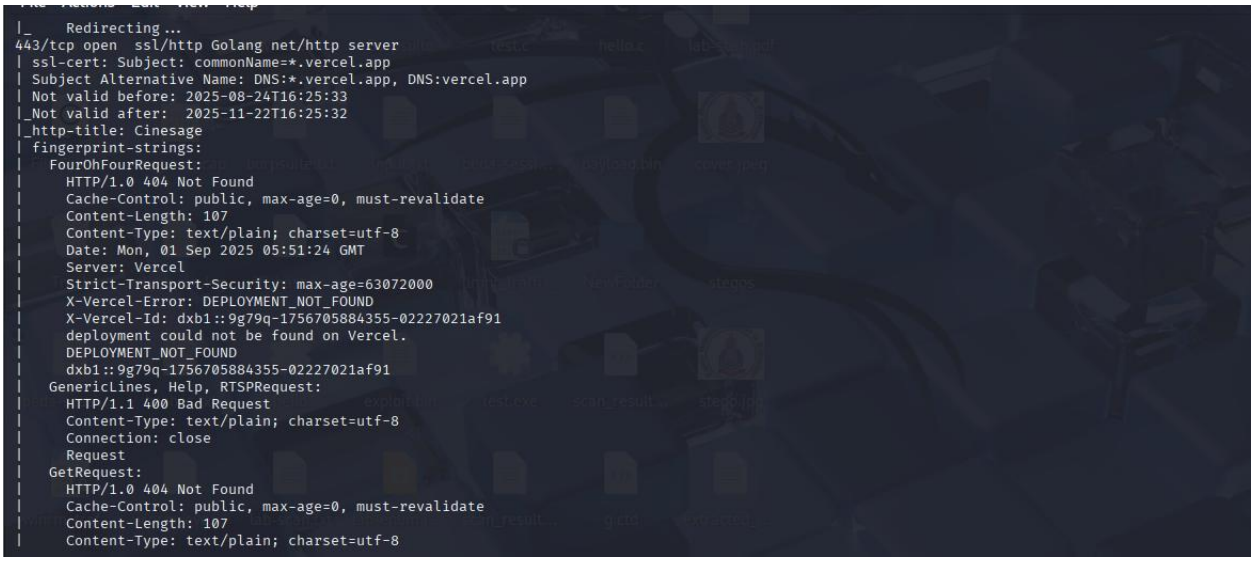
- **Wordlists:** For dirb, specify another list: dirb http://$TARGET /usr/share/wordlists/dirb/common.txt -o ....

- **WhatWeb:** Fast tech fingerprint: whatweb -a 1 http://$TARGET | tee "$OUTPUT_DIR/whatweb.txt".

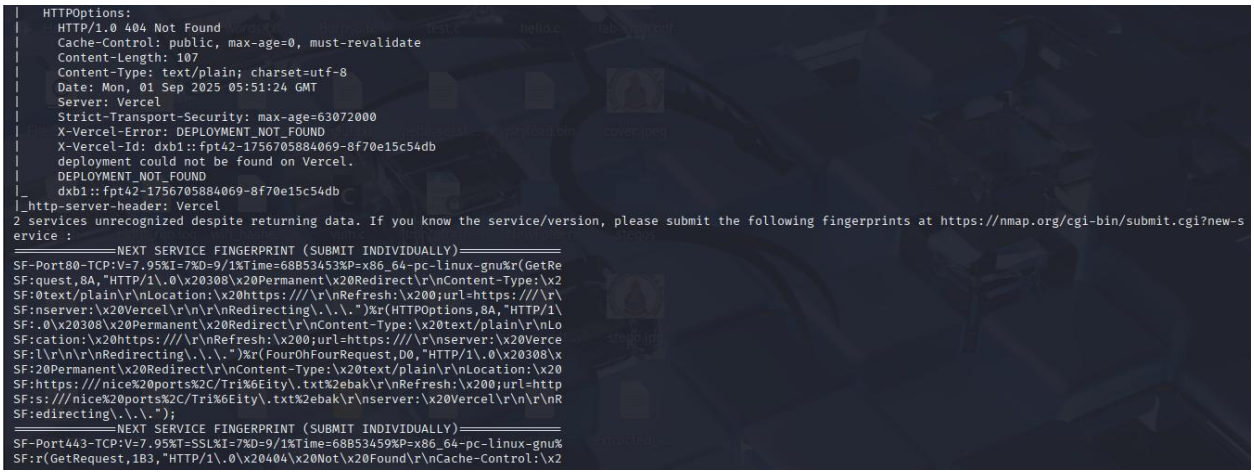
- **Nmap XML parsing:** Use xsltproc to render HTML from XML (nmap -oX then xsltproc).

- **Archive:** Zip everything: tar -czf "$OUTPUT_DIR.tgz" -C "$OUTPUT_DIR" ..

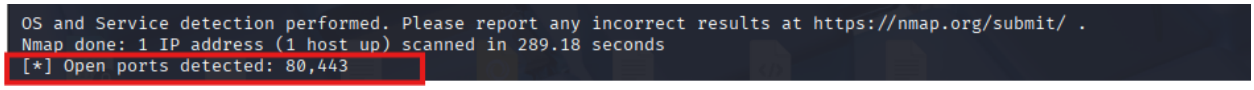




```
|_    Redirecting...
443/tcp open  ssl/http Golang net/http server
| ssl-cert: Subject: commonName=*.vercel.app
| Subject Alternative Name: DNS:*.vercel.app, DNS:vercel.app
| Not valid before: 2025-08-24T16:25:33
|_Not valid after:  2025-11-22T16:25:32
|_http-title: Cinesage
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 404 Not Found
|     Cache-Control: public, max-age=0, must-revalidate
|     Content-Length: 107
|     Content-Type: text/plain; charset=utf-8
|     Date: Mon, 01 Sep 2025 05:51:24 GMT
|     Server: Vercel
|     Strict-Transport-Security: max-age=63072000
|     X-Vercel-Error: DEPLOYMENT_NOT_FOUND
|     X-Vercel-Id: dxb1::9g79q-1756705884355-02227021af91
|     deployment could not be found on Vercel.
|     DEPLOYMENT_NOT_FOUND
|     dxb1::9g79q-1756705884355-02227021af91
|   GenericLines, Help, RTSPRequest:
|     HTTP/1.1 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     Connection: close
|     Request
|   GetRequest:
|     HTTP/1.0 404 Not Found
|     Cache-Control: public, max-age=0, must-revalidate
|     Content-Length: 107
|     Content-Type: text/plain; charset=utf-8
```



```
|   HTTPOptions:
|     HTTP/1.0 404 Not Found
|     Cache-Control: public, max-age=0, must-revalidate
|     Content-Length: 107
|     Content-Type: text/plain; charset=utf-8
|     Date: Mon, 01 Sep 2025 05:51:24 GMT
|     Server: Vercel
|     Strict-Transport-Security: max-age=63072000
|     X-Vercel-Error: DEPLOYMENT_NOT_FOUND
|     X-Vercel-Id: dxb1::fpt42-1756705884069-8f70e15c54db
|     deployment could not be found on Vercel.
|     DEPLOYMENT_NOT_FOUND
|_    dxb1::fpt42-1756705884069-8f70e15c54db
|_http-server-header: Vercel
2 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints at https://nmap.org/cgi-bin/submit.cgi?new-s
ervice :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port80-TCP:V=7.95%I=7%D=9/1%Time=68B53453%P=x86_64-pc-linux-gnu%r(GetRe
SF:quest,8A,"HTTP/1\.0\x20308\x20Permanent\x20Redirect\r\nContent-Type:\x2
SF:0text/plain\r\nLocation:\x20https:///\r\nRefresh:\x200;url=https:///\r\
SF:nserver:\x20Vercel\r\n\r\nRedirecting\.\.\.")%r(HTTPOptions,8A,"HTTP/1\
SF:.0\x20308\x20Permanent\x20Redirect\r\nContent-Type:\x20text/plain\r\nLo
SF:cation:\x20https:///\r\nRefresh:\x200;url=https:///\r\nserver:\x20Verce
SF:l\r\n\r\nRedirecting\.\.\.")%r(FourOhFourRequest,D0,"HTTP/1\.0\x20308\x
SF:20Permanent\x20Redirect\r\nContent-Type:\x20text/plain\r\nLocation:\x20
SF:https:///nice%20ports%2C/Tri%6Eity\.txt%2ebak\r\nRefresh:\x200;url=http
SF:s:///nice%20ports%2C/Tri%6Eity\.txt%2ebak\r\nserver:\x20Vercel\r\n\r\nR
SF:edirecting\.\.\.");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port443-TCP:V=7.95%T=SSL%I=7%D=9/1%Time=68B53459%P=x86_64-pc-linux-gnu%
SF:r(GetRequest,1B3,"HTTP/1\.0\x20404\x20Not\x20Found\r\nCache-Control:\x2
```



```
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 289.18 seconds
[*] Open ports detected: 80,443
```

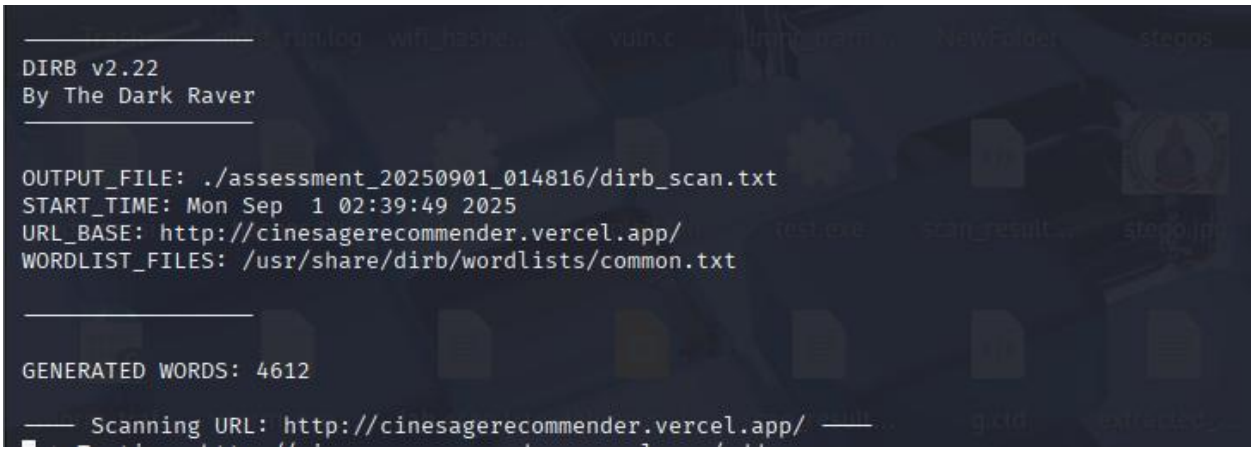
```
[*] Running Nmap vulnerability scan on detected ports...
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-01 01:53 EDT
Nmap scan report for cinesagerecommender.vercel.app (64.29.17.131)
Host is up (0.013s latency).
Other addresses for cinesagerecommender.vercel.app (not scanned): 216.198.79.131

PORT    STATE SERVICE
80/tcp  open  http
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-vuln-cve2014-3704: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| http-dombased-xss: Couldn't find any DOM based XSS.
443/tcp open  https
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-enum:
|   /webtrends/: Potentially interesting folder
|   /web_usage/: Potentially interesting folder
|   /win2k/: Potentially interesting folder
|   /window/: Potentially interesting folder
|   /windows/: Potentially interesting folder
|   /win/: Potentially interesting folder
|   /winnt/: Potentially interesting folder
|   /word/: Potentially interesting folder
|   /work/: Potentially interesting folder
|   /world/: Potentially interesting folder
|   /wsdocs/: Potentially interesting folder
|   /WS_FTP/: Potentially interesting folder
|   /wstats/: Potentially interesting folder
|   /wusage/: Potentially interesting folder
|   /www0/: Potentially interesting folder
```

```
|   /sitecore%20modules/staging/service/api.asmx: Sitecore.NET (CMS)
|   /sitecore%20modules/staging/workdir: Sitecore.NET (CMS)
|_  /sitecore/system/Settings/Security/Profiles: Sitecore.NET (CMS)
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-slowloris-check:
|   VULNERABLE:
|   Slowloris DOS attack
|     State: LIKELY VULNERABLE
|     IDs:  CVE:CVE-2007-6750
|       Slowloris tries to keep many connections to the target web server open and hold
|       them open as long as possible.  It accomplishes this by opening connections to
|       the target web server and sending a partial request. By doing so, it starves
|       the http server's resources causing Denial Of Service.
|
|     Disclosure date: 2009-09-17
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|       http://ha.ckers.org/slowloris/
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to debug)

Nmap done: 1 IP address (1 host up) scanned in 636.82 seconds
```

```
[*] Web service detected. Running Nikto, Dirb, and WPScan...
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Multiple IPs found: 64.29.17.195, 216.198.79.195
+ Target IP:          64.29.17.195
+ Target Hostname:    cinesagerecommender.vercel.app
+ Target Port:        80
+ Start Time:         2025-09-01 02:03:43 (GMT-4)
---------------------------------------------------------------------------
+ Server: Vercel
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Uncommon header 'refresh' found, with contents: 0;url=https://cinesagerecommender.vercel.app/.
+ /: 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: htt
ps://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Root page / redirects to: https://cinesagerecommender.vercel.app/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /database.jks: IP address found in the 'x-vercel-id' header. The IP is "b1::8". See: https://portswigger.net/kb/issues/00600300_private-ip-addresses-disclosed
+ /database.jks: Uncommon header 'x-vercel-id' found, with contents: dxb1::8m2x9-1756706854187-b1851b206e40.
+ /file/../../../../../../../../etc/: Uncommon header 'x-vercel-error' found, with contents: BAD_REQUEST.
+ ERROR: Error limit (20) reached for host, giving up. Last error:
+ Scan terminated: 5 error(s) and 6 item(s) reported on remote host
+ End Time:           2025-09-01 02:39:49 (GMT-4) (2166 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```



TARGET=cinesagerecommender.vercel.app

OUTPUT_DIR=~/assessments/results_$(date +%Y%m%d_%H%M%S)

mkdir -p "$OUTPUT_DIR"

echo "Running Nikto..."

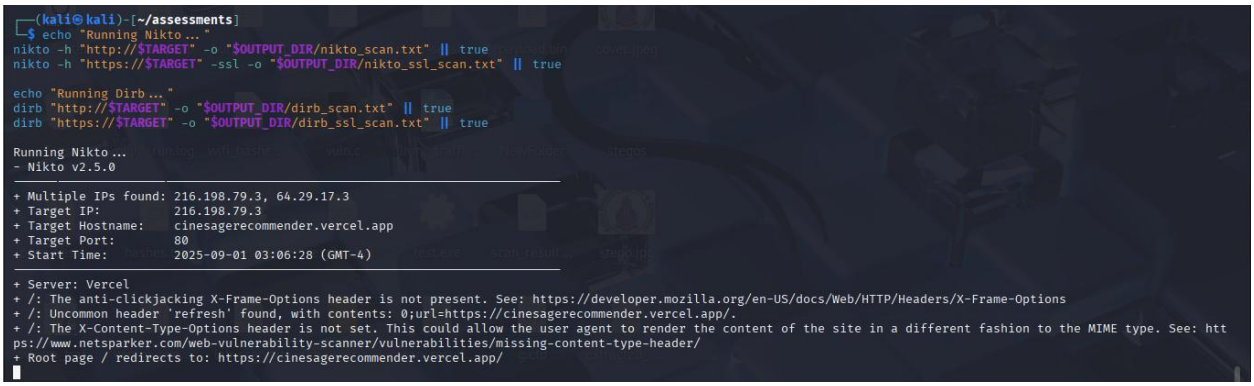
nikto -h "http://$TARGET" -o "$OUTPUT_DIR/nikto_scan.txt" || true

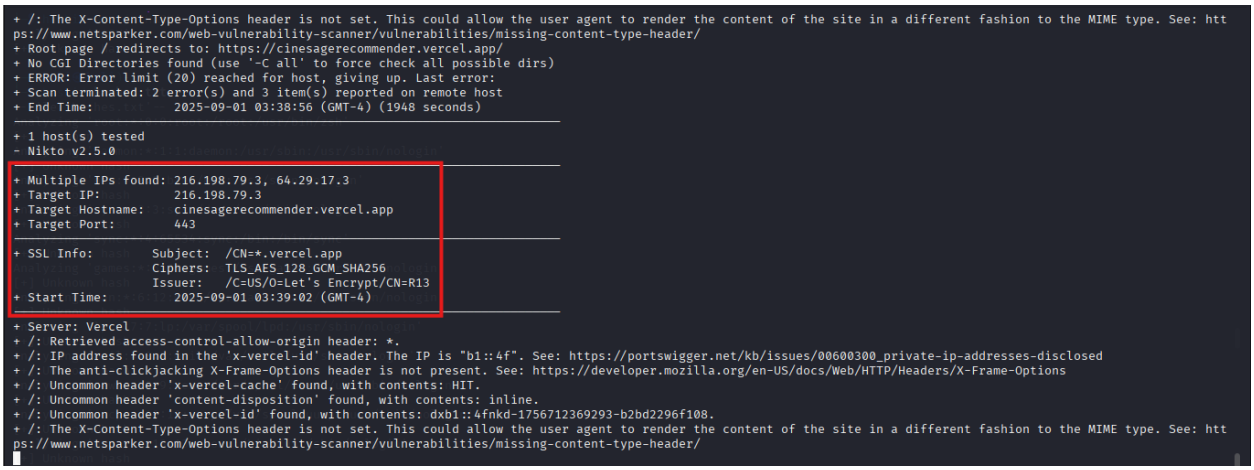
nikto -h "https://$TARGET" -ssl -o "$OUTPUT_DIR/nikto_ssl_scan.txt" || true

echo "Running Dirb..."

dirb "http://$TARGET" -o "$OUTPUT_DIR/dirb_scan.txt" || true

dirb "https://$TARGET" -o "$OUTPUT_DIR/dirb_ssl_scan.txt" || true





7) Troubleshooting

- **No ports found:** Host firewalled; try -Pn, slower timing, or confirm target/route.

- **Nikto/Dirb time out:** Service may be HTTPS-only or behind WAF; try https:// explicitly.

- **WPScan missing:** Install/update: sudo apt install wpscan or gem install wpscan.
- **Permission errors:** Re-run with sudo (some scans require elevated privileges).

- **Report empty:** Check run.log for errors and confirm files exist in the output dir.

9) Safety checklist (print & tape near your monitor)

☐ Do I have written authorization (or is this a designated lab box)?

☐ Are my scan timings conservative for this environment?

☐ Am I capturing outputs with timestamps and not overwriting previous runs?

☐ Did I store results in an approved, access-controlled folder?

☐ Did I verify any “vulnerable” findings manually before reporting?

Appendix — One-liners you can reuse

```
# Quick open-port list from Nmap XML using awk
awk -F '"' '/portid=/ && /state state="open"/ {print $4}' nmap_scan.xml | paste -sd, -

# Safer baseline Nmap
nmap -sS -sV -sC -T3 -oA baseline <target>

# Render Nmap XML to HTML (requires xsltproc)
xsltproc /usr/share/nmap/nmap.xsl nmap_scan.xml > nmap_scan.html

# Quick WordPress check (cheap)
curl -ksS http://<target> | grep -qi 'wp-content\|wp-includes' && echo "WordPress likely"
```

End of manual.