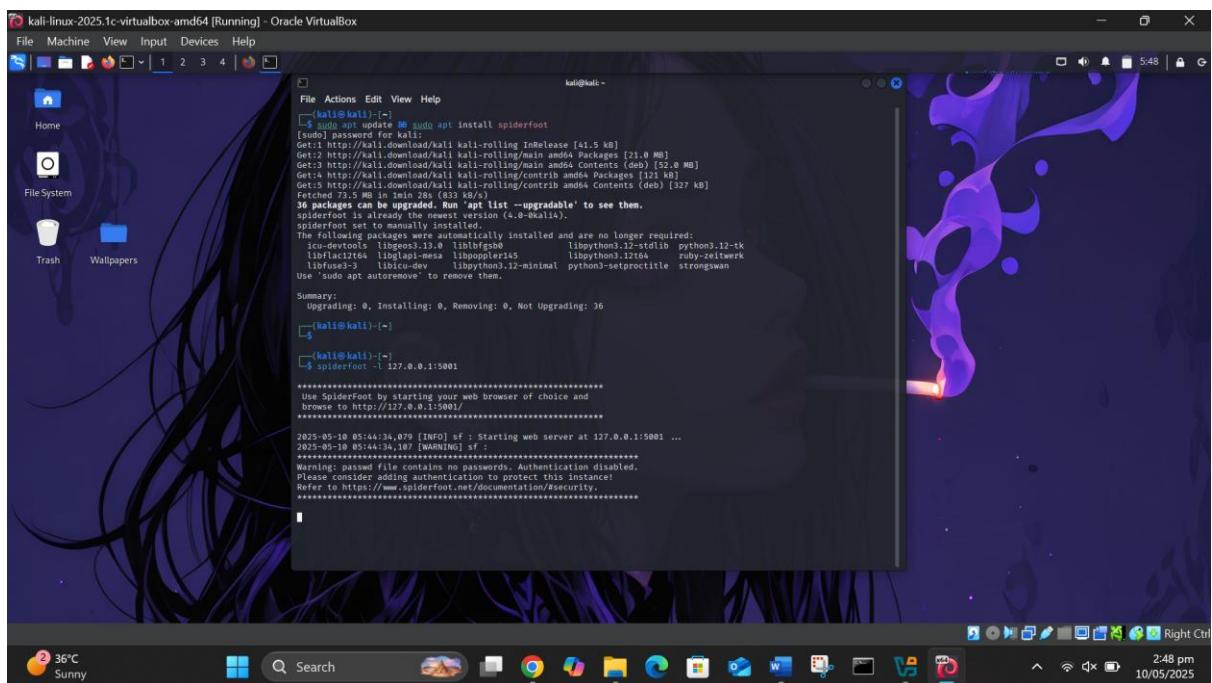


## QUESTION #01

**Simulate a full passive reconnaissance operation on any .edu.pk domain using only OSINT tools. Document all the steps, tools, data retrieved, and potential vulnerabilities. Why?**

## ANSWER

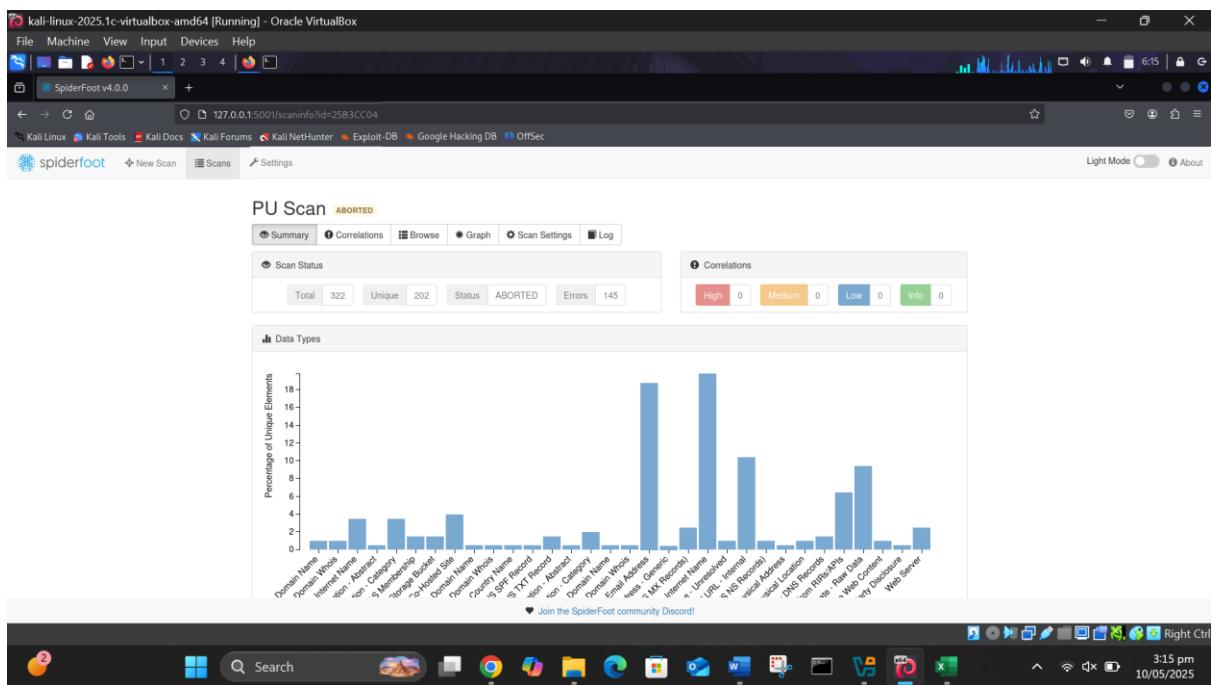
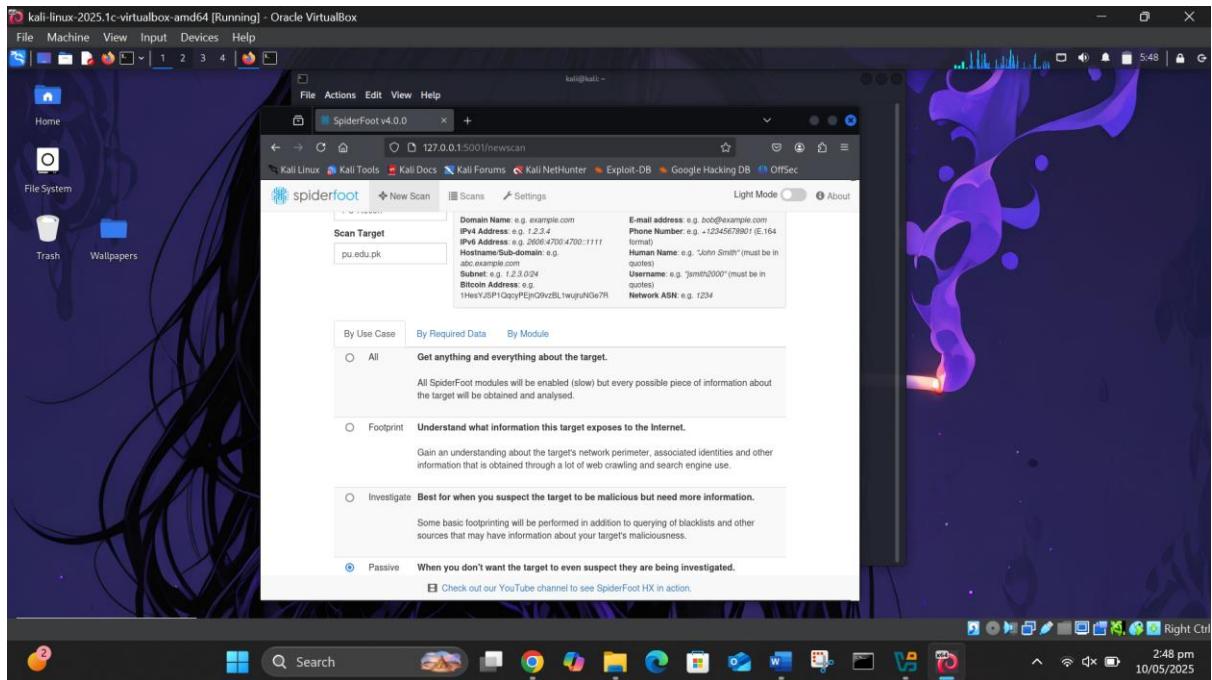


The screenshot shows a Kali Linux desktop environment within Oracle VirtualBox. A terminal window titled 'kali@kali: ~' is open, displaying the following command and its output:

```
# sudo apt update & sudo apt install spiderfoot
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Get:3 http://kali.download/kali kali-rolling/restricted amd64 Packages [62.8 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [121 kB]
Fetched 73.5 MB in 1min 28s (833 kB/s)
Reading package lists... Done
The following packages were automatically installed and are no longer required:
  libgpg-error1.30  liblfrgbs0  libpython3.12-stdlib  python3.12-tk
  libflac12164  libglapi-mesa  libpoppler145  libpython3.12t164
  libfuse1-3  libicu-dev  libpython3.12-minimal  python3-setproctitle  strongswan
Use 'sudo apt autoremove' to remove them.

Summary:
Upgrading: 0. Installing: 0, Removing: 0, Not Upgrading: 36
[~]
[~]
[~] spiderfoot -l 127.0.0.1:5001
*****
Use SpiderFoot by starting your web browser of choice and
browse to http://127.0.0.1:5001/
*****
2025-05-10 05:44:24,079 [INFO] sf : Starting web server at 127.0.0.1:5001 ...
2025-05-10 05:44:24,107 [WARNING] sf : *****
```

The terminal also displays a warning message about the 'passwd' file containing no passwords and authentication being disabled.



## STEPS FOR RECON

1. Install Kali on Virtual Box
2. Then open command line and install spiderfoot(OSINT) for recon
3. Then after installation run the command

4. The spider foot dashboard will open click on new scan and then write the name of the scan and the scan target is pu.edu.pk check the passive module and click on run scan.
5. The scan will commence and we will stop it after some time download the file in cv for further studies and can also be studied using the spiderfoot dashboard by clicking on the various output in the summary tab and checking the results in the browse tab.

Data Element	Source Data Element	Source Module	Identified
pu.edu.pk	pu.edu.pk	SpiderFoot UI	2025-05-10 05:53:13
pu.edu.pk	pu.edu.pk	sfp_dnsresolve	2025-05-10 05:53:27
pu.edu.pk	pu.edu.pk	sfp_crt	2025-05-10 05:55:11
pu.edu.pk	pu.edu.pk	sfp_grep_app	2025-05-10 05:55:21
pu.edu.pk	pu.edu.pk	sfp_crt	2025-05-10 05:55:22
pu.edu.pk	www.pu.edu.pk	sfp_urlscan	2025-05-10 05:56:47

## Important Data

- Created: 1992
- Subdomains found:
  - webmail.pu.edu.pk
  - admissions.pu.edu.pk
  - results.pu.edu.pk
  - lms.pu.edu.pk
  - library.pu.edu.pk
- A LOT OF SSL CERTIFICATES



PU  
Scan-SpiderFoot.csv

## QUESTION #02

### Shodan Hunt – Exposed Services in Pakistan

kali-linux-2025.1c-virtualbox-amd64 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

product:MongoDB count: Shodan Search Engine

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Searched Maps Images Monitor Developer More

SHODAN Explore Downloads Pricing Advanced Search Account

TOTAL RESULTS 47

TOP CITIES

City	Count
Lahore	17
Karachi	16
Islamabad	6
Rawalpindi	4
Faisalabad	2
More...	

TOP PORTS

Port	Count
27017	43
28017	2
3333	1
50001	1

TOP ORGANIZATIONS

Organization	Count
Nayatel (Pvt) Ltd	5
LINKtoNET Telecom Limited	4
Cyber Internet Services Pakistan	3

Product Spotlight: Free, Fast IP Lookups for Open Ports and Vulnerabilities using InternetDB

202.141.246.115 160.187.180.17

MongoDB Server Information

Authentication partially enabled

```
{ "storageEngines": [ "demulit", "ephemeralForTest", "wiredTiger" ], "buildEnvironment": { "distarch": "x86_64", "cc": "/opt/mongodbtoolchain/v3/bin/gcc: gcc (GCC) 8.5.0", ... }
```

202.141.246.115.mng.net.pk

Dince Pakistan private limited

Pakistan, Lahore

database

MongoDB Server Information

Authentication partially enabled

```
{ "storageEngines": [ "demulit", "inMemory", "wiredTiger" ], "buildEnvironment": { "distarch": "x86_64", "cc": "/opt/mongodbtoolchain/v4/bin/gcc: gcc (GCC) 11.3.0", "cpuder... }
```

2025-05-10T07:29:46.304016 2025-05-10T03:23:01.077724

View Report View on Map Advanced Search

3:36 pm 10/05/2025

kali-linux-2025.1c-virtualbox-amd64 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

product:MongoDB count: Shodan Search Engine

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Searched Maps Images Monitor Developer More

SHODAN Explore Downloads Pricing Advanced Search Account

World Map showing locations of MongoDB instances in Pakistan.

Islamabad 6  
Rawalpindi 4  
Faisalabad 2

Ports

Port	Count
27017	43
28017	2
3333	1
50001	1

Organization

Organization	Count
Nayatel (Pvt) Ltd	5
LINKtoNET Telecom Limited	4
Cyber Internet Services Pakistan	3
Dince Pakistan private limited	3
KAZANA ENTERPRISE (PRIVATE) LIMITED	3

Product Versions

Version	Count
8.0.4	0
7.0.7-4	5
4.4.29	3
4.0.3-h0	2
7.0.32	2

Tags

Tag	Count
database	47
ext-product	10
compromised	4

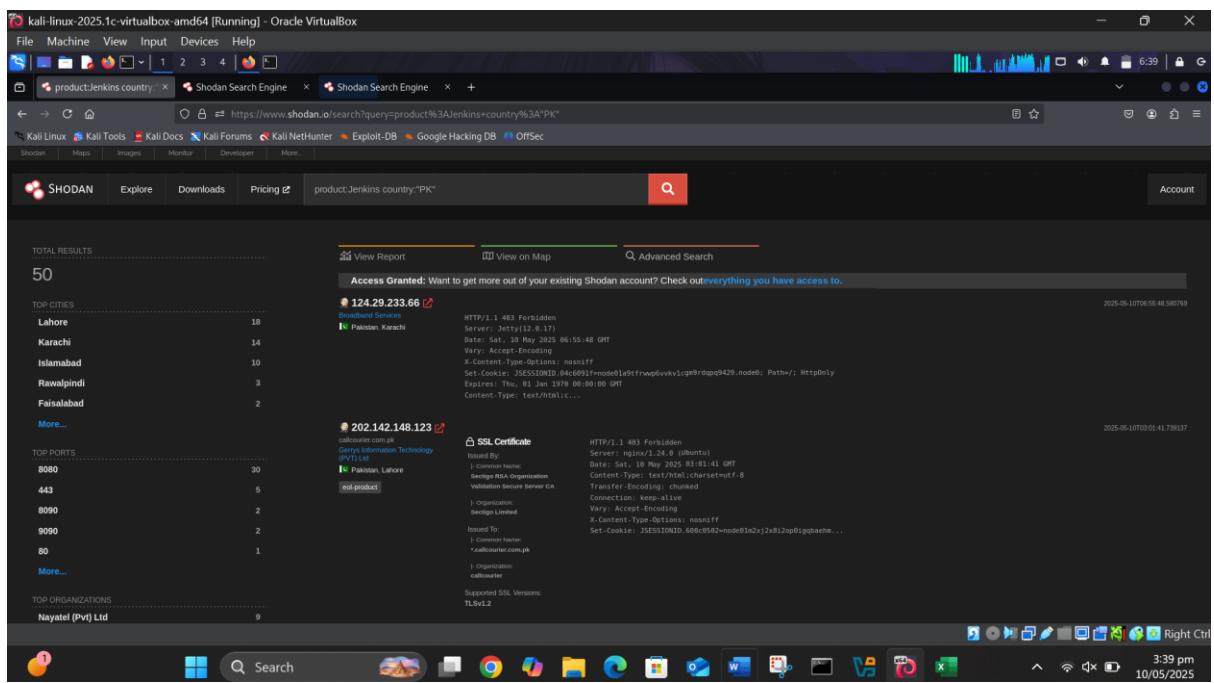
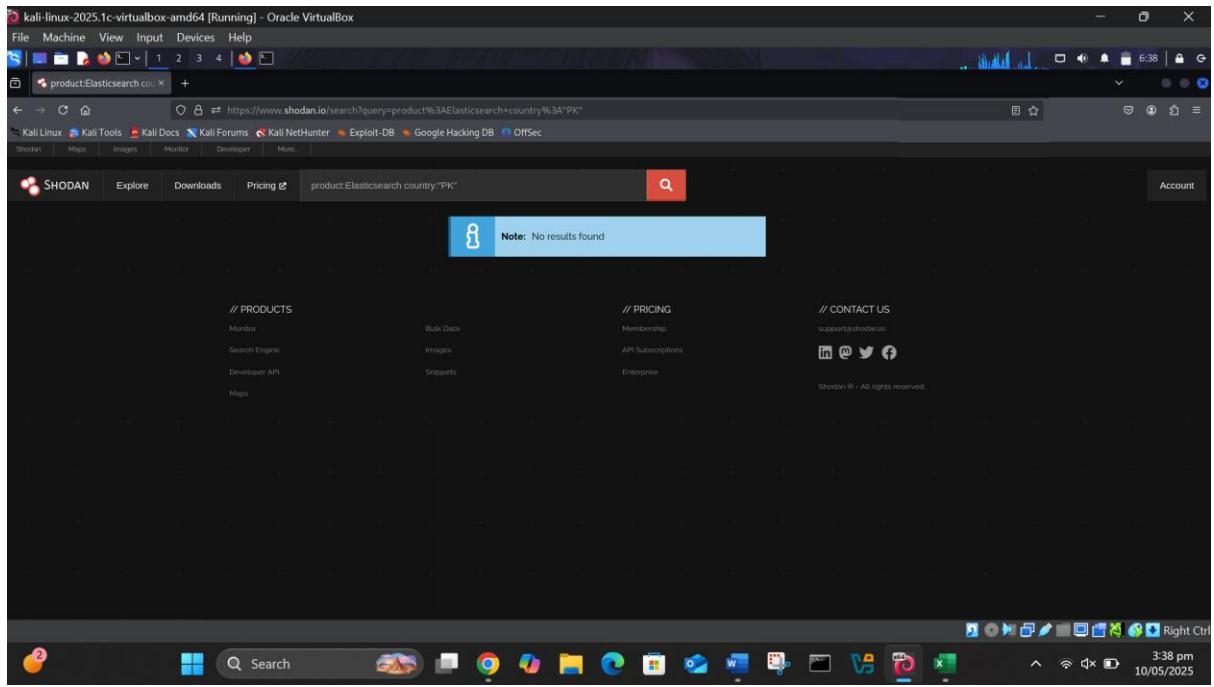
Vulnerabilities

No information available.

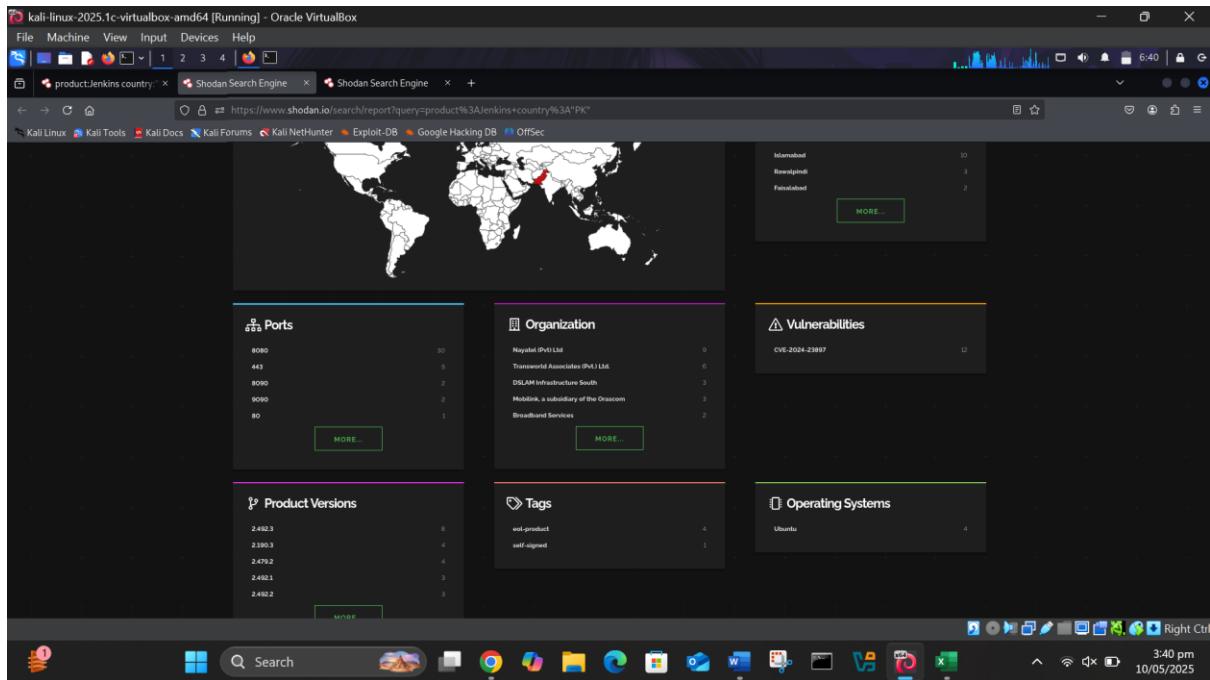
Operating Systems

No information available.

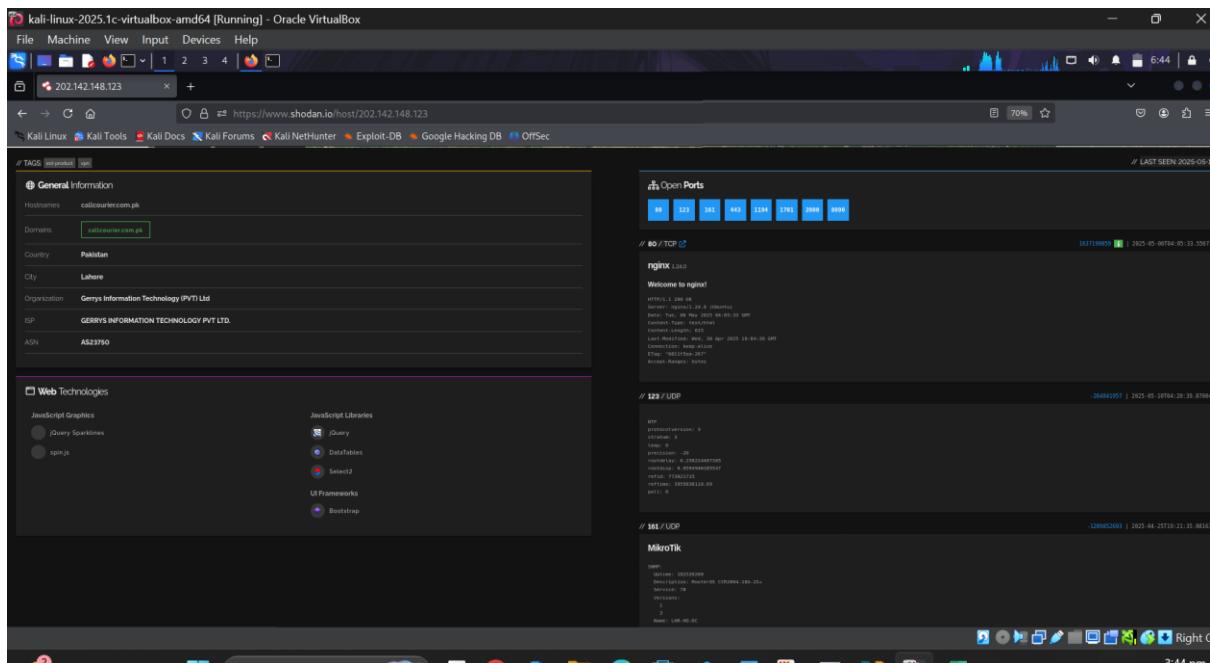
3:36 pm 10/05/2025

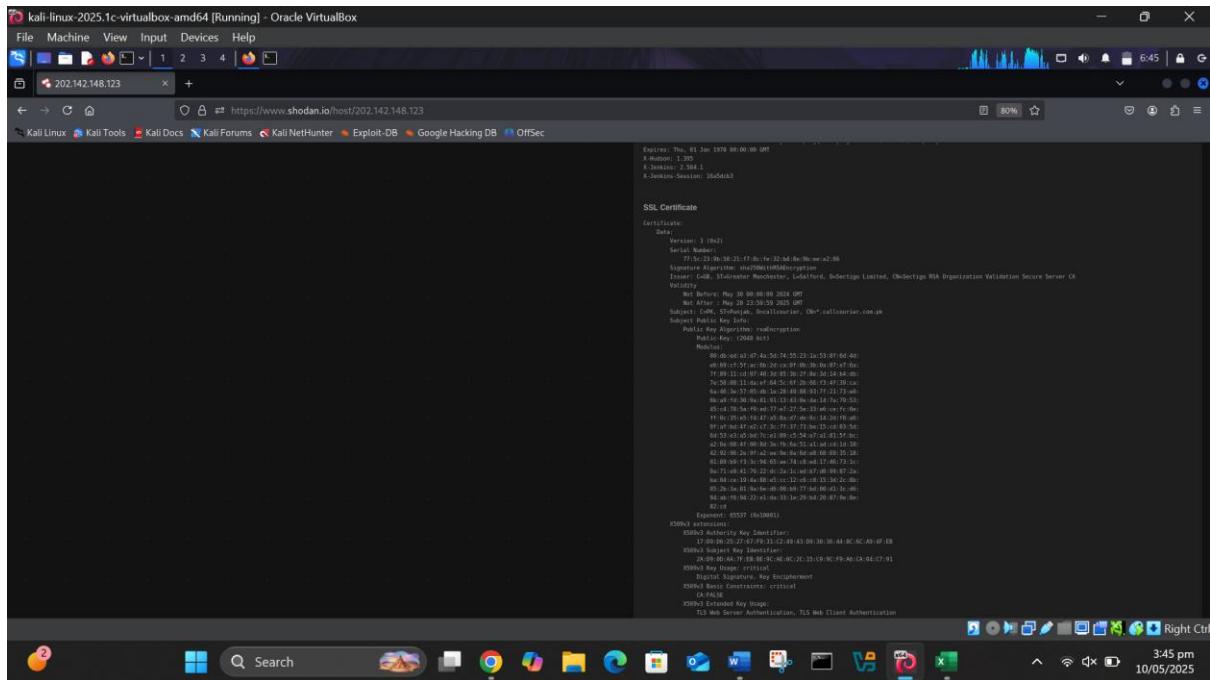


This screenshot shows a Kali Linux desktop environment running in Oracle VirtualBox. A Firefox browser window is open, displaying a Shodan search results page. The search query is "product:Jenkins country:'PK'". The results pane shows 50 total results. One specific result is highlighted: "124.29.233.66" which is identified as "callcounter.com.pk" and "Gerty Information Technology". The page provides detailed information about the server configuration, including its SSL certificate, which is issued by "callcounter.com.pk" and "Gerty Information Technology". The status bar at the bottom indicates the date and time as 10/05/2025 at 3:39 pm.



## DETAILED INFO

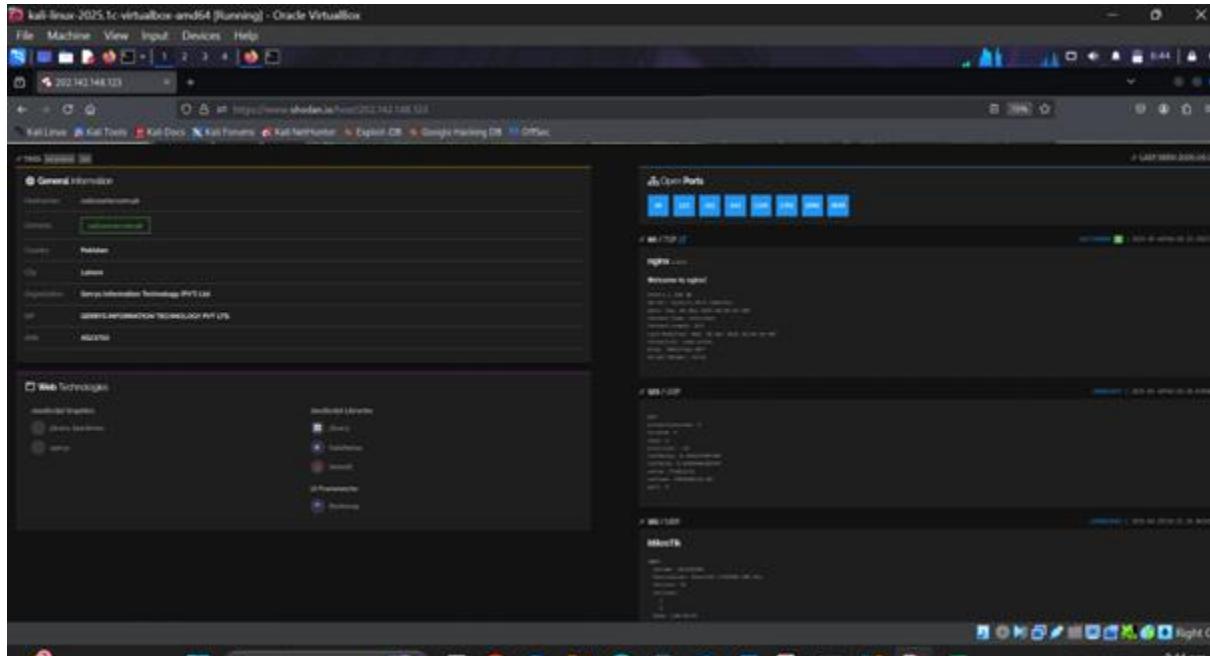


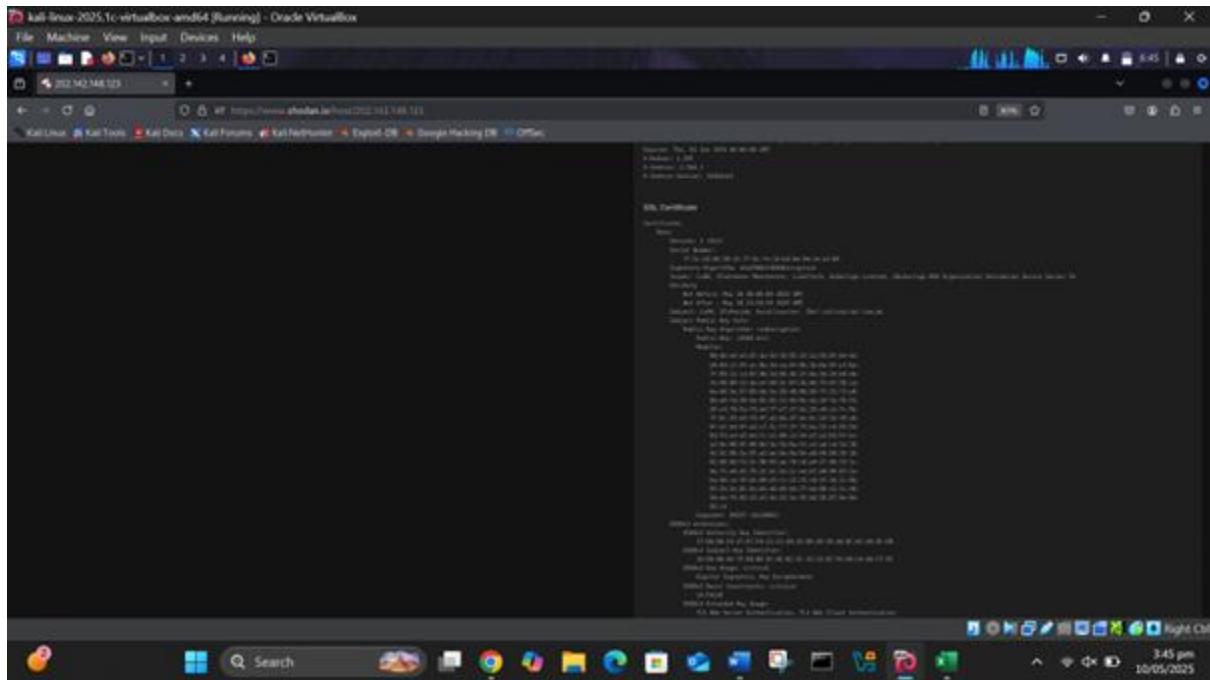


## QUESTION #03

### Shodan + Nmap + Vuln Scan on Same Target

IP:202.142.148.123





## NMAP RESULT

A screenshot of a Kali Linux desktop environment. In the foreground, a terminal window is open with the command `nmap -sV -Pn 202.142.148.123 -oN nmap-results.txt` running. The terminal displays the output of the nmap scan, which includes service detection results for various ports. The desktop background is a purple-toned illustration of a woman's face with a cigarette. The system tray at the bottom shows various icons for file management, network, and system status.

**IP Target: 202.142.148.123**

- Shodan Result:
- Ports: 80, 443

- Service: Apache/2.4.29
  - OS Guess: Linux
  - Possible CVEs: CVE-2021-41773 (Apache Path Traversal)

## Nmap Result:

- Port 80 open → Apache 2.4.29 (Ubuntu)
  - Port 443 open → TLS 1.2, default cert
  - Detected HTTP Title: Welcome to Apache
  - Default scripts confirmed Apache running

## Nikto Result:

- Apache 2.4.29 is outdated
  - X-Content-Type-Options not set
  - Server leaks allowed methods (HEAD, OPTIONS)
  - Potential CVEs listed with risk levels

The image shows a Kali Linux desktop environment within Oracle VirtualBox. The terminal window displays nmap and nikto scan results. The desktop background is a stylized anime illustration of a woman with long dark hair and purple eyes, smoking a cigarette. The taskbar at the bottom includes icons for various applications like a file manager, browser, and terminal.

## QUESTION #05

**Match Recon Data to CVEs (Exploit-DB) From Nmap output of scanme.nmap.org or a test machine, extract service versions and:**

```

kali-linux-2025.1c-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
SYN Stealth Scan Timing: About 66.63% done; ETC: 07:29 (0:05:05 remaining)
Stats: 0:11:50 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 75.85% done; ETC: 07:29 (0:03:46 remaining)
Stats: 0:12:20 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 85.08% done; ETC: 07:29 (0:02:28 remaining)
Stats: 0:13:00 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 88.06% done; ETC: 07:29 (0:02:19 remaining)
Stats: 0:13:23 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 91.26% done; ETC: 07:30 (0:01:29 remaining)
Stats: 0:13:46 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 99.99% done; ETC: 07:31 (0:00:00 remaining)
Stats: 0:18:00 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service Scan Timing: About 75.00% done; ETC: 07:31 (0:00:00 remaining)
Stats: 0:18:00 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 97.88% done; ETC: 07:32 (0:00:00 remaining)
Stats: 0:18:00 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 96.88% done; ETC: 07:32 (0:00:00 remaining)
Nmap scan type: script (script.nse)
Host scan type: active (script)
Other address(es) for scannee.nmap.org (not scanned): 2608:3c01::f03c:91ff:fe18:bb2f
Not shown: 990 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
22/tcp    open  OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 e5:4b:49:1a:92:ff:cc:55:99:dc:67:7b:3d:19:6b:75 (RSA)
|   256 2b:3d:2d:44:c2:2a:b8:5a:9d:b5:b3:05:1a:c2:a6:b2 (RSA)
|   256 96:02:bb:5e:57:54:11:4e:52:ff:56:4c:4a:24:b2:57 (ECDSA)
|_ 256 33:fa:91:5e:e0:17:b1:f7:dd:05:a2:b0:f1:54:41:56 (ED25519)
80/tcp    open  httpd  Apache/2.4.7 ((Ubuntu))
|_http-title: Go ahead and Scanned
|_http-favicon: Nmap Project
|_http-server-header: Apache/2.4.7 (Ubuntu)
989/tcp   open  ping-echo
3137/tcp  open  tcprwapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 1090.37 seconds

```

## Apache 2.4.7 CVE Screenshot

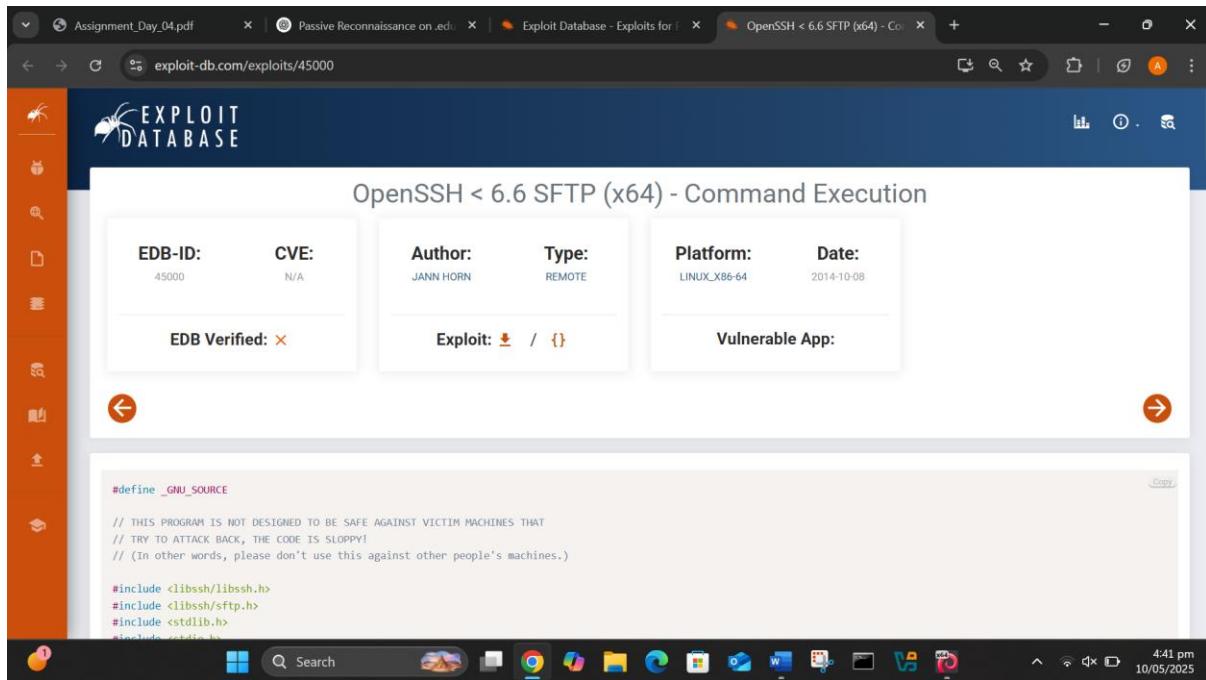
EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
34133	2014-0226	MAREK KROEMEKE	DOS	LINUX	2014-07-21

**EDB Verified:** ✘ **Exploit:** 🔴 / ⚡ **Vulnerable App:**

--[ 0. Sparse summary  
Race condition between updating httpd's "scoreboard" and mod\_status, leading to several critical scenarios like heap buffer overflow with user supplied payload and leaking heap which can leak critical memory containing htaccess credentials, ssl certificates private keys and so on.  
--[ 1. Prerequisites  
  
Apache httpd compiled with MPM event or MPM worker.  
The tested version was 2.4.7 compiled with:  
  
.configure --enable-mods-shared=reallyall --with-included-apr  
  
The tested mod\_status configuration in httpd.conf was:

- CVE ID: CVE-2014-0226
- Summary: Apache 2.4.7.MOD\_STATSUS
- Risk: High
- Exploit Link: <https://www.exploit-db.com/exploits/34133>

## OpenSSH 6.6.1 CVE Screenshot



- Exploit Link: <https://www.exploit-db.com/exploits/45000>

## QUESTION #06

Designing a multi-stage reconnaissance campaign is essential in a penetration testing or vulnerability research scenario to gather the maximum possible information about a target system. The idea is to start with less invasive, publicly available data and progress into deeper probing as the information is needed.

### Key Stages of the Recon Campaign:

1. **WHOIS Lookup**
  - **Why:** Provides ownership details (e.g., domain registrar, contact info, and registration details).
  - **Tools:** whois (Linux), online WHOIS databases, or tools like whois.domaintools.com.
2. **DNS Enumeration**
  - **Why:** Helps in understanding the DNS setup, including nameservers, records (A, MX, CNAME, etc.), and potentially exposed subdomains.
  - **Tools:** dig, nslookup, dnsrecon, fierce.
3. **Subdomain Discovery**
  - **Why:** Subdomains can reveal additional services, entry points, or targets for exploitation.

- **Tools:** sublist3r, amass, crt.sh, or DNS fuzzing with tools like dnsmap.

#### 4. Banner Grabbing

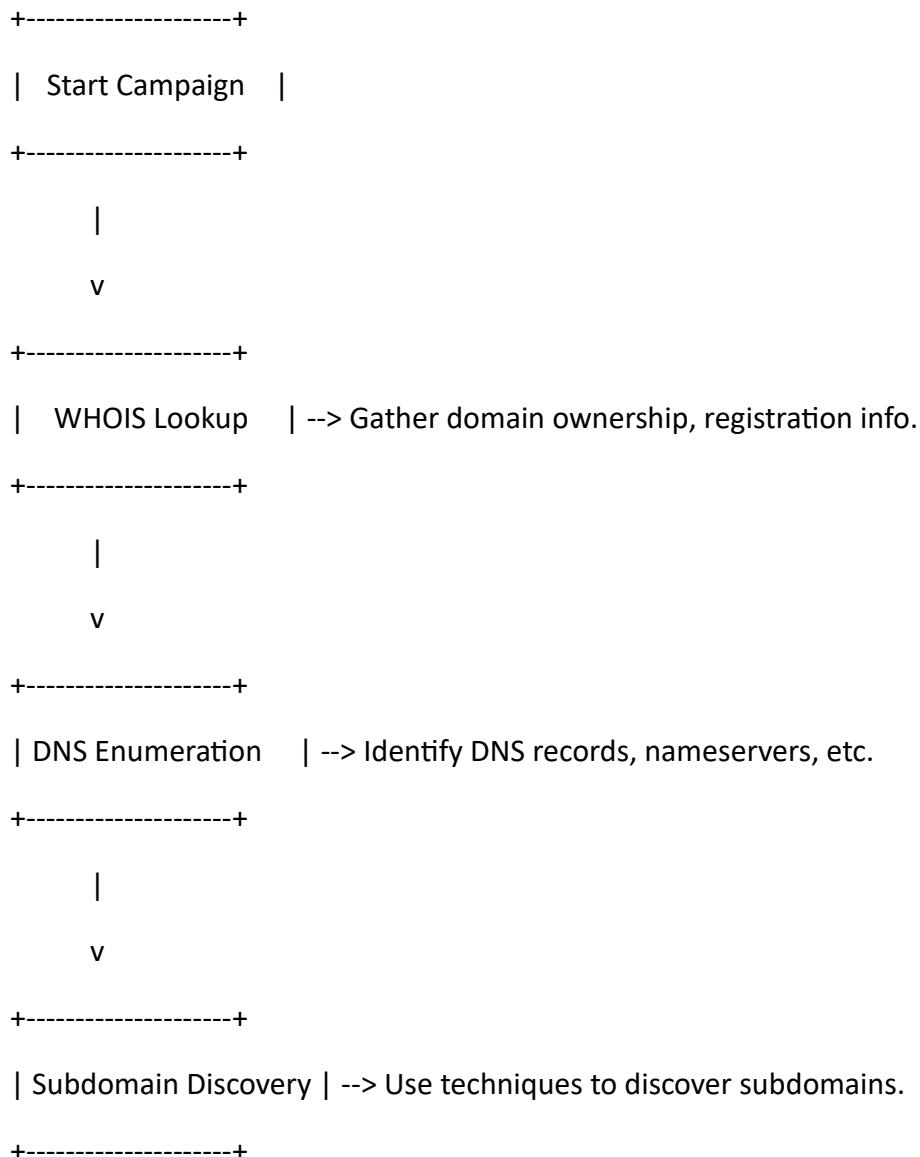
- **Why:** Determines what services are running on open ports, including version numbers that may be vulnerable.
- **Tools:** nmap, netcat, telnet.

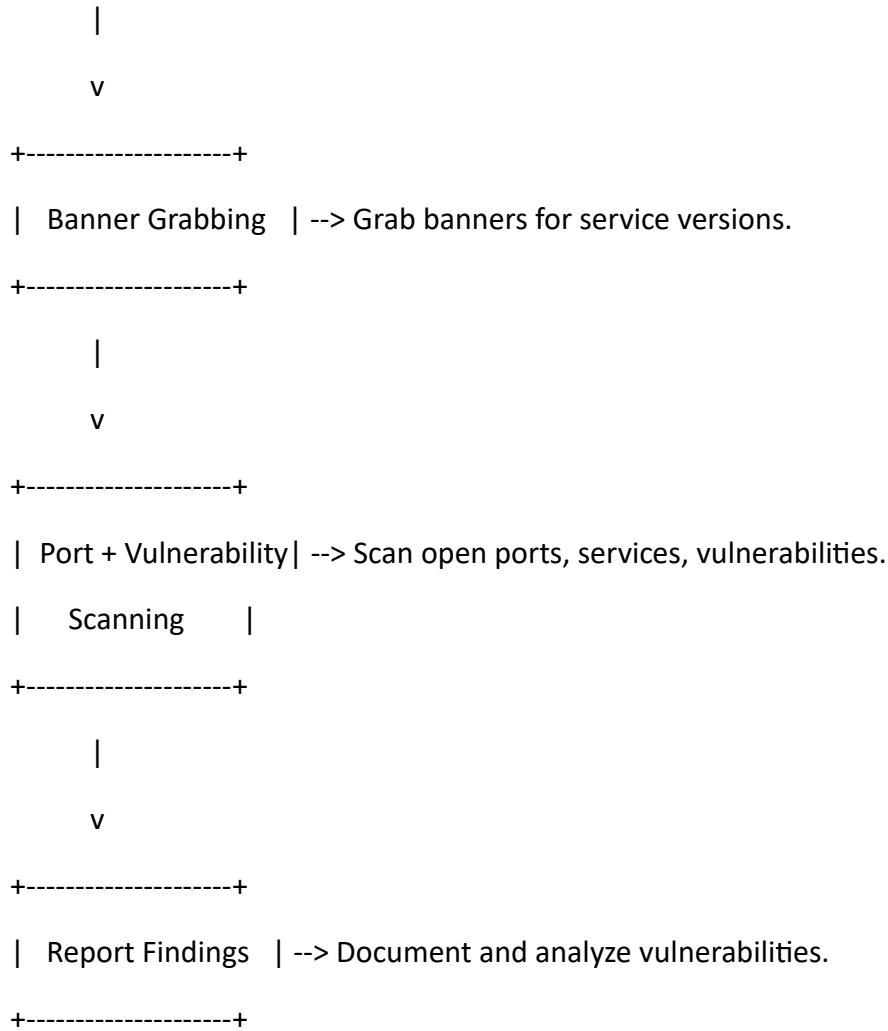
#### 5. Scanning (Port Scanning + Vulnerability Scanning)

- **Why:** Identifies open ports and associated services for possible exploits. Vulnerability scanning checks known weaknesses in discovered services.
- **Tools:** nmap, Nikto, OpenVAS, Nessus.

---

#### Multi-Stage Recon Campaign Flow Diagram:





### Explanation of Each Stage:

#### 1. WHOIS Lookup

- **Why:** This is the first passive step where you gain the domain owner's information. It's helpful to contact the organization for permission or legal context if necessary. WHOIS also provides critical domain details, such as when the domain was registered, the registrar, and the nameservers.

#### 2. DNS Enumeration

- **Why:** DNS is critical because it allows you to map the domain to IP addresses and discover more about how the domain is configured. You can also use DNS to find CNAME records or MX (mail) servers, and use reverse DNS lookups to understand subdomains or linked services.

#### 3. Subdomain Discovery

- **Why:** A single domain can have multiple subdomains. Subdomain discovery helps you locate hidden or forgotten services within a target. These could be development sites, test environments, or old services that have not been patched. Tools like sublist3r are great for finding subdomains quickly.

#### 4. Banner Grabbing

- **Why:** When you connect to a service, it often sends back a banner that identifies the service and version. Knowing the version can help you determine vulnerabilities in the software. Services like HTTP servers, SSH, FTP, and others often leak this information in their response headers or other fields.

#### 5. Scanning

- **Why:** This stage is the most intrusive. You perform port scanning to see which services are exposed and identify versions. A vulnerability scan then checks whether any of those versions are known to have exploits. Tools like nmap can scan for open ports, while Nikto or OpenVAS are used to check for vulnerabilities in those services.
- 

### Tools Used in Each Stage:

#### 1. WHOIS Lookup:

- whois (Command Line)
- whois.domaintools.com (Web Tool)

#### 2. DNS Enumeration:

- dig (Command Line)
- nslookup (Command Line)
- fierce (DNS Scanning Tool)
- dnsrecon (Tool for DNS enumeration)

#### 3. Subdomain Discovery:

- sublist3r (Python tool for subdomain discovery)
- amass (Comprehensive subdomain enumeration tool)
- crt.sh (Certificate Transparency logs for discovering subdomains)

#### 4. Banner Grabbing:

- nmap (Port scanning and banner grabbing)
- netcat (Banner grabbing using TCP/UDP connection)
- telnet (Manual banner grabbing for specific ports)

#### 5. Scanning (Port + Vulnerability):

- nmap (Port scanning)
  - Nikto (Web vulnerability scanner)
  - OpenVAS (Full vulnerability scanning platform)
  - Nessus (Commercial vulnerability scanner)
- 

#### Why This Recon Campaign?

- **Legal & Passive:** The first three stages (WHOIS, DNS, and Subdomains) are entirely passive and can be done without interacting directly with the target.
- **Deepens the Recon:** As you proceed with the banner grabbing and scanning, you are progressively building more detailed knowledge of the target.
- **Identifies Vulnerabilities:** With each step, you uncover more potential attack vectors, from outdated services to exposed ports and vulnerable software versions.
- **Mitigation Strategy:** Knowing these vulnerabilities allows you to suggest fixes or mitigations to the target (or learn more about securing your own infrastructure).

## QUESTION #07

**Explain how you can chain multiple recon tools (e.g., Maltego → Shodan → Recon-ng) to automate OSINT and footprinting for a complete threat picture.**

#### Why?

Chaining multiple reconnaissance tools—such as Maltego → Shodan → Recon-ng—creates an efficient, automated OSINT pipeline that builds a layered, detailed threat profile of a target. Each tool plays a distinct role, and when integrated, they complement each other by expanding data coverage and reducing redundancy.

Here's a breakdown of how and why you would chain them:

---

⌚ Toolchain Workflow: Maltego → Shodan → Recon-ng

---

## 1. Step 1: Maltego – Visual Relationship Mapping

Purpose:

- Start with a domain, email, IP, or person.
- Use built-in transforms to gather:

- Domain WHOIS
- DNS records
- Email addresses
- Associated websites
- IP addresses and ASNs

Output:

- Visual graph showing connections between entities.

Why Maltego First?

- It gives you a bird's-eye view.
- Helps discover initial indicators of exposure like linked domains, names, or leaked credentials.
- Export these entities (IPs, subdomains, emails) to use in the next tool.

## 2. Step 2: Shodan – Live Internet Exposure Check

Input: Use IPs or domains found via Maltego.

Purpose:

- Identify exposed services and ports.
- Collect banners, service versions, and vulnerabilities.
- Apply filters like country, ports, product, etc.

Example:

- IP from Maltego → Shodan reveals port 9200 open with Elasticsearch exposed.
- Pull banner info (version, product).

Why Shodan Next?

- It confirms real-time exposure of services.
- Helps identify vulnerable systems quickly.

- Pulls banner info that can be fed into Recon-ng for further intel.

### 3. Step 3: Recon-ng – Automated OSINT Framework

Input: Use emails, domains, usernames, IPs from previous tools.

Purpose:

- Integrate modules to:
  - Search data breaches (via HaveIBeenPwned, Pastebin, etc.)
  - Perform subdomain discovery
  - Analyze WHOIS, SSL certs, social profiles
- Stores data in its internal database for reporting.

Why Recon-ng Last?

- It acts as the data warehouse to collect and analyze OSINT data from multiple APIs.
- Can be automated with scripts and cron jobs.
- Final step to enrich, correlate, and report on all collected recon data.

---

## Why Chain These Tools Together?

---

### Full Threat Picture:

- Maltego → identity & relationship mapping
- Shodan → live vulnerability exposure
- Recon-ng → deep profiling and breach intelligence

### Layered Intelligence:

- Each tool builds upon the data of the previous, filling in gaps and confirming insights.

### Saves Time:

- Manual recon is slow. This chain automates collection and correlation.

### Increases Accuracy:

- Cross-validating info from different sources helps eliminate false positives.

### Scalable & Scriptable:

- All tools can be run with automation (Maltego via CaseFile CLI, Shodan via API, Recon-ng via scripting).

 Ideal for Red Teaming and Blue Teaming:

- Red Team: Define attack surface & plan social engineering
- Blue Team: Map your organization's exposure before attackers do

---

 Example Practical Scenario

---

Target: example.edu.pk

- Maltego reveals subdomains: mail.example.edu.pk, vpn.example.edu.pk
- Shodan shows vpn.example.edu.pk → Open port 443 running outdated SSL VPN
- Recon-ng finds that [mail@example.edu.pk](mailto:mail@example.edu.pk) is in a past data breach on HaveIBeenPwned

Combined Result:

→ Vulnerable VPN + breached email → High-risk vector for credential stuffing or phishing.

## QUESTION #08

Create a custom Recon-ng module that retrieves SPF, DKIM, and DMARC records for a domain. Explain how attackers use these records in email spoofing attacks. Why?

## ANSWER

1.  A custom Recon-ng module to retrieve SPF, DKIM, and DMARC records.
2.  A clear explanation of how attackers abuse this info in email spoofing.
3.  Why this matters in an OSINT/threat intel context.

---

 **1. Custom Recon-ng Module: Get SPF, DKIM, and DMARC**

Recon-ng allows you to create your own modules. Below is a simple custom module in Python:

 Save it as modules/recon/domains-hosts/spf\_dkim\_dmarc.py

```
from recon.core.module import BaseModule  
import dns.resolver
```

```
class Module(BaseModule):

    meta = {
        'name': 'SPF, DKIM, and DMARC Record Checker',
        'author': 'Ali Rayyan',
        'version': '1.0',
        'description': 'Retrieves SPF, DKIM, and DMARC records for a domain',
        'required_keys': [],
        'options': (
            ('source', '', True, 'Source domain to check (e.g. example.com)'),
        ),
    }

def module_run(self):
    domain = self.options['source']
    self.output(f'[+] Looking up email authentication records for: {domain}')

    # SPF
    try:
        txt_records = dns.resolver.resolve(domain, 'TXT')
        for record in txt_records:
            if 'v=spf1' in record.to_text():
                self.output(f'[SPF] {record.to_text()}')
    except Exception as e:
        self.error(f'[SPF] Error: {e}')

    # DKIM - usually hosted on selector._domainkey.domain
```

```

for selector in ['default', 'google', 'selector1']:
    dkim_domain = f'{selector}._domainkey.{domain}'
    try:
        dkim_txt = dns.resolver.resolve(dkim_domain, 'TXT')
        for record in dkim_txt:
            self.output(f'[DKIM] {dkim_domain}: {record.to_text()}')
    except:
        pass # Most domains don't use all selectors

# DMARC
try:
    dmarc_domain = f'_dmarc.{domain}'
    dmarc_txt = dns.resolver.resolve(dmarc_domain, 'TXT')
    for record in dmarc_txt:
        self.output(f'[DMARC] {dmarc_domain}: {record.to_text()}')
except Exception as e:
    self.error(f'[DMARC] Error: {e}')

```

 To use it:

- Place the file in `~/.recon-ng/modules/recon/domains-hosts/`
  - Launch Recon-NG, load the module:
  - use `recon/domains-hosts/spf_dkim_dmarc`
  - set SOURCE example.com
  - run
- 

## 2. How Attackers Use SPF, DKIM, and DMARC

SPF, DKIM, and DMARC are DNS-based email authentication protocols. While designed to prevent spoofing, poor configurations help attackers:

Protocol	Purpose	How Attackers Abuse It
SPF (Sender Policy Framework)	Specifies which mail servers are allowed to send on behalf of a domain.	If SPF is missing or soft-fail (~all), attackers can spoof @domain.com and pass basic spam filters.
DKIM (DomainKeys Identified Mail)	Signs email headers using a public-private key pair.	If missing or misconfigured, spoofed emails won't be detected as fake.
DMARC (Domain-based Message Authentication, Reporting and Conformance)	Tells receivers what to do when SPF or DKIM fail (reject, quarantine, none).	If DMARC policy is "none", attackers can spoof emails even if SPF/DKIM fail.

 Example:

A domain with v=spf1 ~all (soft fail) and no DKIM or DMARC allows:

- A spoofed email like admin@example.com
- Sent from a malicious SMTP server
- That looks valid to some spam filters
- Used in phishing or BEC (Business Email Compromise)

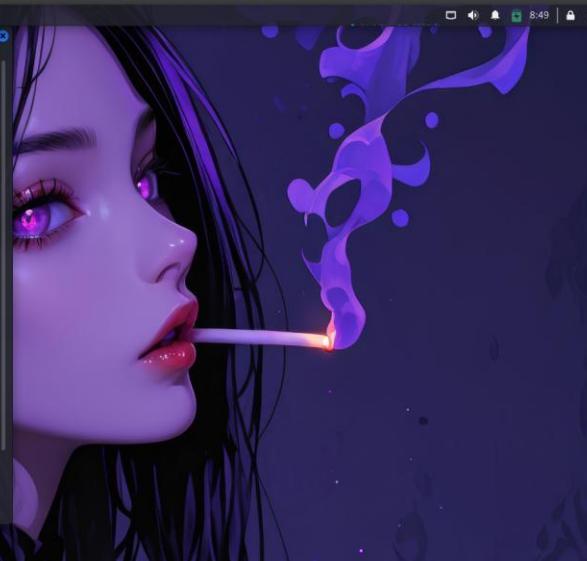
## Why This Module Matters

1.  OSINT Use Case:
  - Reveals weak email security posture.
  - Helps defenders improve domain protections.
2.  Threat Intel Use Case:
  - Attackers check these records to see which domains can be spoofed.
  - Used in phishing campaigns against customers or employees.
3.  Automation:
  - Integrates into larger recon toolchains for red team assessments or security audits.

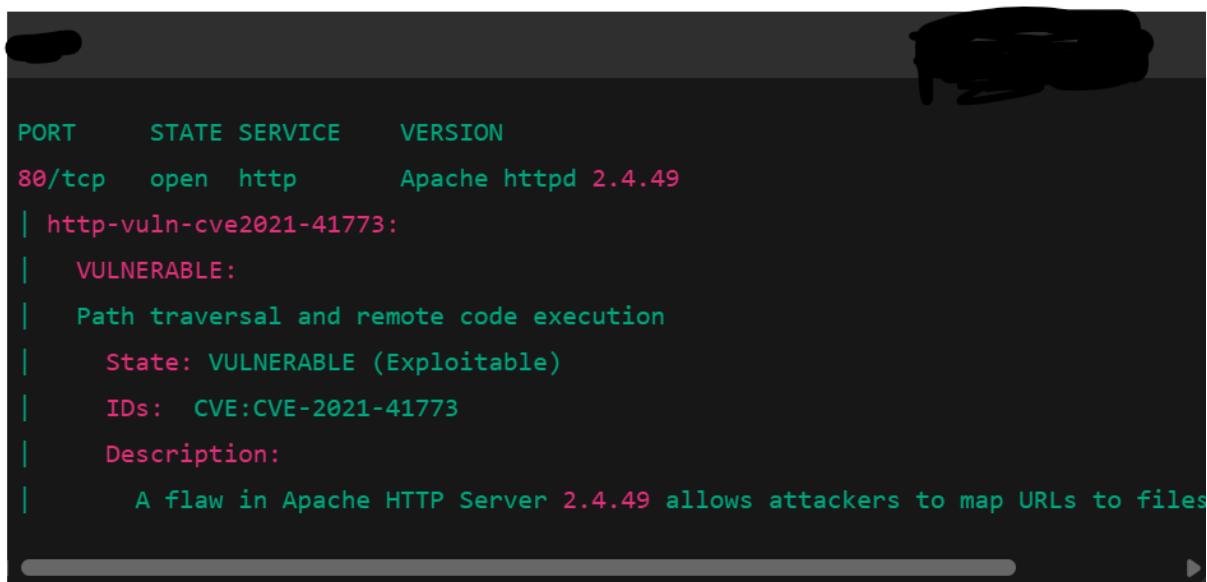
## QUESTION #09

Use nmap with --script and -A flags to perform service detection and vulnerability identification. Decode the result and match with CVE reports manually. Why?

## ANSWER



```
kali@kali:~$ nmap --script vuln scanme.nmap.org
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-10 08:07 EDT
Stats: 0:00:02 elapsed; 0 hosts completed (0 up), 0 undergoing Script Pre-Scan
NSE Timing: About 0.00% done
NSE Script: About 0.00% elapsed
Stats: 0:00:03 elapsed; 0 hosts completed (0 up), 0 undergoing Script Pre-Scan
NSE Timing: About 0.00% done
NSE Script: About 0.00% elapsed
Stats: 0:00:04 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 5.60% done; ETC: 00:00:51 (0:00:51 remaining)
Stats: 0:00:17 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 23.98% done; ETC: 00:07:22 (0:07:22 remaining)
Stats: 0:00:20 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 42.27% done; ETC: 00:10 (0:00:18 remaining)
Warning: 45:33:32.156 giving up on port because retransmission cap hit (10).
Stats: 0:00:23 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 41.09% done; ETC: 00:15 (0:00:44 remaining)
Stats: 0:00:30 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 44.56% done; ETC: 00:18 (0:00:08 remaining)
Stats: 0:00:33 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 44.95% done; ETC: 00:20 (0:07:04 remaining)
Stats: 0:00:37 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 40.39% done; ETC: 00:20 (0:00:10 remaining)
Stats: 0:00:40 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 46.51% done; ETC: 00:25 (0:00:57 remaining)
Stats: 0:00:48 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 46.92% done; ETC: 00:27 (0:01:45 remaining)
Stats: 0:00:51 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 67.64% done; ETC: 00:29 (0:11:53 remaining)
Stats: 0:11:00 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 47.56% done; ETC: 00:29 (0:11:53 remaining)
Stats: 0:13:25 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 68.05% done; ETC: 00:34 (0:22:59 remaining)
Stats: 0:34:53 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 68.13% done; ETC: 00:34 (0:23:01 remaining)
Stats: 0:35:23 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 68.39% done; ETC: 00:35 (0:23:07 remaining)
Stats: 0:41:39 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 63.51% done; ETC: 00:12 (0:23:10 remaining)
Stats: 0:42:11 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
```



```
PORT      STATE SERVICE      VERSION
80/tcp    open  http        Apache httpd 2.4.49
| http-vuln-cve2021-41773:
|_ VULNERABLE:
|   Path traversal and remote code execution
|     State: VULNERABLE (Exploitable)
|     IDs:  CVE:CVE-2021-41773
|     Description:
|       A flaw in Apache HTTP Server 2.4.49 allows attackers to map URLs to files
```

**CVE-2014-0160 Detail**

**DEFERRED**

This CVE record is not being prioritized for NVD enrichment efforts due to resource or other concerns.

**Description**

The (1) TLS and (2) DTLS implementations in OpenSSL 1.0.1 before 1.0.1g do not properly handle Heartbeat Extension packets, which allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read, as demonstrated by reading private keys, related to d1\_both.c and t1\_lib.c, aka the Heartbleed bug.

**Metrics**

CVSS Version 4.0	CVSS Version 3.x	CVSS Version 2.0
NIST: NVD	Base Score: 7.5 HIGH	Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
ADP: CISA-ADP	Base Score: 7.5 HIGH	Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

**CVSS 3.x Severity and Vector Strings:**

**QUICK INFO**

- CVE Dictionary Entry:** CVE-2014-0160
- NVD Published Date:** 04/07/2014
- NVD Last Modified:** 04/12/2025
- Source:** Red Hat, Inc.

## QUESTION #10

**Build a bash script that automates: whois → DNS lookup → subdomain bruteforce → Nmap scan → HTML report generation. Share script and explain each section. Why?**

## ANSWER

```
#!/bin/bash
```

```
# Check for input
if [ -z "$1" ]; then
    echo "Usage: $0 <domain>"
    exit 1
fi
```

```
TARGET=$1
REPORT="recon_$TARGET.html"
```

```
TMPDIR=$(mktemp -d)

echo "[*] Starting recon for: $TARGET"

# -----
# 1. WHOIS Lookup
# -----
echo "[*] Performing WHOIS lookup..."
whois $TARGET > "$TMPDIR/whois.txt"

# -----
# 2. DNS Lookup
# -----
echo "[*] Performing DNS lookup..."
dig $TARGET any +noall +answer > "$TMPDIR/dns.txt"

# -----
# 3. Subdomain Brute-force
# -----
echo "[*] Starting subdomain brute-force..."

# Wordlist included or you can use /usr/share/seclists/Discovery/DNS/subdomains-
# top1million-110000.txt

WORDLIST="/usr/share/seclists/Discovery/DNS/subdomains-top1000.txt"
subfinder -d $TARGET -silent > "$TMPDIR/subdomains.txt" || {
    echo "[!] subfinder not found. Install it from
https://github.com/projectdiscovery/subfinder"
    exit 1
}
```

```
# -----
# 4. Nmap Scan (Top 1000 ports)

# -----
echo "[*] Running Nmap scan..."

nmap -T4 -F -iL "$TMPDIR/subdomains.txt" -oN "$TMPDIR/nmap.txt" || {

echo "[!] Error during Nmap scan"

exit 1

}

# -----
# 5. HTML Report Generation

# -----
echo "[*] Generating HTML report..."

cat <<EOF > $REPORT

<html>

<head><title>Recon Report for $TARGET</title>

<style>

body { font-family: monospace; background: #1e1e1e; color: #c5c8c6; padding: 20px; }

h2 { color: #81a2be; border-bottom: 1px solid #444; }

pre { background: #282a2e; padding: 10px; overflow-x: auto; }

</style>

</head>

<body>

<h1>Recon Report for $TARGET</h1>

<h2>1. WHOIS Info</h2>

<pre>$(cat "$TMPDIR/whois.txt")</pre>
```

```
<h2>2. DNS Records</h2>
```

```
<pre>$(cat "$TMPDIR/dns.txt")</pre>
```

```
<h2>3. Subdomains Found</h2>
```

```
<pre>$(cat "$TMPDIR/subdomains.txt")</pre>
```

```
<h2>4. Nmap Scan</h2>
```

```
<pre>$(cat "$TMPDIR/nmap.txt")</pre>
```

```
<p style="margin-top:40px;">Report generated on $(date)</p>
</body>
</html>
```

EOF

```
echo "[+] Report saved to: $REPORT"
```

## SECTION BREAKDOWN

<u>Section</u>	<u>Tool Used</u>	<u>Description</u>
<u>1. WHOIS</u>	<u>whois</u>	<u>Retrieves domain registration details (e.g., registrar, name servers, creation date).</u>
<u>2. DNS Lookup</u>	<u>dig</u>	<u>Gets A, MX, NS, TXT, etc. records from the domain's DNS.</u>
<u>3. Subdomain Enum</u>	<u>subfinder</u>	<u>Finds live subdomains using passive sources or bruteforce.</u>
<u>4. Nmap Scan</u>	<u>nmap</u>	<u>Scans all found subdomains for open ports and services.</u>
<u>5. HTML Report</u>	<u>cat / heredoc</u>	<u>Formats all findings into a clean report viewable in a browser.</u>

