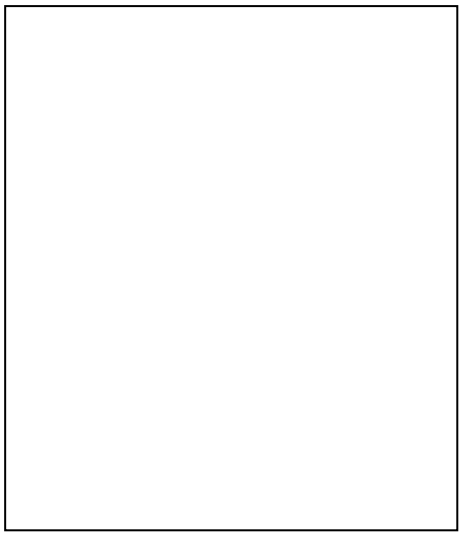
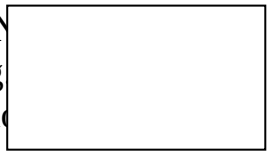
Department of Computer Science & Engineering



Project Name: Web Application Reconnaissance and Vulnerability Detection

Professor
Dept. of Computer Science & Engineering
Islamic University, Bangladesh.

Submitted By
Habibur Rahoman
Roll N
Reg
Sessi



Declaration

I, hereby, declare that the work presented in this project is the outcome of the investigation performed by us under the supervisi r, Department of Computer Science & Engineering, declare that the project has been developed with integrity and adherence to ethical standards. By signing below, I affirm my dedication to delivering a project on **Web Application Reconnaissance and Vulnerability Detection** that meets ethical standards and fulfills user expectations.

_____

Signature

Habibur Rahoman

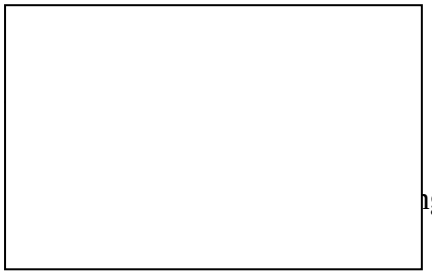
Roll No

Session:

Certification

I am pleased to certify that Habibur Rahoman (E
eted a project titled "**Web Application Reconnaissance and Vulnerability Detection**" under my supervision. This project fulfills the partial requirements for the degree of BSc Engineering.

Certified by:



ngineering

Acknowledgement

o Allah, for successfully completion of
ask requires the effort of many people
o express my sincere gratitude to my
, Department of Computer Science &
also grateful to other teachers of
mic University, Bangladesh in making

e Department of Computer science and Engineering for their cooperation during my studies and those who gave me the possibility to complete this project.

Also I will not forget friends and relatives who supported me in any respect during the completion of the project.

ABSTRACT

Reconnaissance is a critical phase in ethical hacking and penetration testing, where detailed information about a target system is gathered to identify potential vulnerabilities. In the Project, **Recon.sh** is a comprehensive Bash script designed to automate and streamline the reconnaissance process, leveraging a variety of tools to collect, analyze, and organize critical information about a given domain.

This script integrates multiple open-source tools to perform tasks such as subdomain enumeration, live subdomain detection, HTTP service analysis, URL extraction, port scanning, and parameter discovery. Key components include **Subfinder**, **Assetfinder**, **crt.sh**, **HTTPx**, **GAU**, **Waybackurls**, **Nmap**, and **ParamSpider**, making it a versatile solution for both bug bounty hunting and penetration testing.

By automating time-intensive tasks, **Recon.sh** reduces manual effort while improving accuracy and efficiency in data collection. The script outputs organized data, including live subdomains, open ports, archived URLs, and potential attack surfaces, in a structured directory format. This enables security professionals to focus on analyzing the gathered information for potential vulnerabilities.

The script is highly extensible, allowing users to add new tools or modify workflows to suit specific use cases. Ethical considerations, such as ensuring proper authorization and secure data handling, are integral to its intended use. Overall, **Recon.sh** is a valuable tool for security researchers and professionals aiming to enhance their reconnaissance process.

Table of Contents

CHAPTER 1

INTRODUCTION

Automation plays a pivotal role in cybersecurity, especially during the reconnaissance phase of penetration testing and bug bounty hunting. *Recon.sh* is a versatile script designed to automate tasks such as subdomain enumeration, active URL identification, and parameter discovery. However, while automation enhances speed and coverage, manual intervention in vulnerability detection remains essential for a robust and accurate security assessment. Combining both approaches ensures a more comprehensive evaluation of web application security.

1.1 Overview of Automation in Cybersecurity

Automation tools in cybersecurity are critical in enhancing efficiency, consistency, and accuracy in protecting organizations from evolving threats. These tools streamline complex tasks, reduce human error, and help manage the growing volume of data and alerts that security teams handle daily. Below is an overview of key automation tools and their applications in cybersecurity.

1.2 Purpose of Recon.sh

Recon.sh is a Bash script designed to automate the reconnaissance phase of cybersecurity assessments, such as penetration testing, vulnerability scanning, and bug bounty hunting. Its purpose is to simplify and enhance the process of gathering critical information about a target system or network, enabling security professionals to identify potential vulnerabilities efficiently and effectively.

1.3 Key Objectives of Recon.sh

Streamline Reconnaissance Tasks
Automates repetitive and time-consuming tasks like subdomain enumeration, active URL detection, and parameter discovery, saving valuable time and effort.

Comprehensive Data Collection
Integrates multiple tools to ensure a thorough and detailed assessment of the target, consolidating data from various sources.

Organized Output
Organizes results into structured directories and files, making it easier to analyze findings and prepare reports.

Reduce Manual Errors
Minimizes human errors by automating workflows, ensuring more reliable and consistent results.

Customizability and Flexibility
The modular design allows users to modify the script, add new tools, and tailor it to specific use cases or environments.

1.4 Functional Goals

Subdomain Enumeration

- Identify subdomains using tools like Subfinder, Assetfinder, and crt.sh.
- Ensure maximum coverage by leveraging multiple data sources.

Active URL Identification

- Use HTTPx to detect live subdomains and identify accessible endpoints.

Screenshot Capture

- Automate visual documentation of live endpoints using HTTPx or Gowitness.

Historical and Dynamic URL Extraction

- Extract URLs from historical archives using tools like GAU and Waybackurls.

Port Scanning

- Conduct detailed scans of open ports and services using Nmap.

Parameter Discovery

- Use Paramspider to identify URL parameters, which are often targets for injection attacks.

Manual Vulnerability Detection

Perform hands-on testing on selected web applications (e.g., *phpvulnweb*) to manually identify security issues like XSS, SQL Injection, and authentication flaws.
Supports deeper analysis by validating automated findings and discovering logic-based vulnerabilities that automation may overlook.

1.5 Benefits of Recon.sh

Efficiency

Automates the reconnaissance process, allowing professionals to focus on deeper analysis. It will correctly perform all the scanning and getting to the endpoints.

Scalability

Suitable for both individual and enterprise-level assessments. It encompasses the best practices for scaling.

Accuracy and Time-Saving

Enhances reliability by reducing manual errors and validating with manual inspection. It

saves time on repetitive tasks, enabling faster project completion.

Centralized Information

Consolidates outputs into structured directories.

Deeper Insight via Manual Testing

Manual analysis ensures detection of logic flaws and complex vulnerabilities that tools may miss.

1.6 Use Cases

Bug Bounty Programs
Helps bounty hunters efficiently discover common and edge-case vulnerabilities.
Penetration Testing
Assists professionals in performing thorough assessments using both automated and manual techniques.
Proactive Security Monitoring
Organizations can periodically scan and test for vulnerabilities, reducing risk exposure.

CHAPTER 2

PROJECT OBJECTIVES

Project Objectives

The primary objective of a reconnaissance and vulnerability detection project, especially in cybersecurity contexts like bug bounty hunting or penetration testing, is to systematically gather, analyze, and report critical information about a target's attack surface. This information lays the foundation for identifying vulnerabilities and improving the target's security posture. Below are common objectives tailored to reconnaissance.

2.1 Identify the Attack Surface

- **Objective:** Uncover all publicly accessible systems, domains, subdomains, IP addresses, and other exposed assets of the target.
- **Why it's important:** This ensures no critical component of the target's infrastructure is overlooked and helps build a complete picture of potential entry points.

2.2 Gather Technical Insights

Objective: Collect detailed information about the target's technologies, configurations, and services, such as:

- Operating systems and software versions
- Web servers, APIs, and frameworks in use
- Open ports and network services

Why it's important: Understanding the target's technical environment helps identify misconfigurations, outdated software, or known vulnerabilities.

2.3 Discover Vulnerabilities

- **Objective:** Pinpoint potential security weaknesses during the reconnaissance phase without active exploitation, such as:
  - Exposed sensitive files, directories, or credentials
  - Misconfigured DNS or SSL/TLS
  - Vulnerable APIs or endpoints

  Why it's important: Early identification of vulnerabilities allows for a focused approach during exploitation and mitigation phases.

2.4 Maintain Ethical and Legal Compliance

Objective: Ensure all reconnaissance activities are performed within legal and ethical boundaries, including:

- Staying within the authorized scope
- Respecting user privacy and data protection regulations

Why it's important: Ethical reconnaissance protects the reputation of the security professional and avoids legal consequences.

2.5 Create Actionable Reports

Objective: Generate clear, detailed, and actionable reports that highlight:

- Identified assets
- Potential vulnerabilities
- Recommendations for remediation

Why it's important: A well-documented report enables stakeholders to address security gaps effectively and strengthens their security posture.

2.6 Enhance Operational Efficiency

Objective: Optimize reconnaissance workflows to reduce time and effort while ensuring comprehensive coverage by:

- Automating repetitive tasks like subdomain enumeration or port scanning
- Using tools like Recon.sh, Nmap, or Amass efficiently

Why it's important: Automation allows security professionals to focus on analysis and critical tasks, increasing productivity.

2.7 Vulnerability and Penetration Testing

Objective: Set the stage for advanced security assessments like penetration testing by:

- Highlighting the most vulnerable or exposed components
- Identifying high-risk systems for further testing

Why it's important: Reconnaissance serves as a precursor to deeper assessments, improving the overall effectiveness of the security testing process.

2.8 Support Bug Bounty and Penetration Testing Goals

Objective: Assist bug bounty hunters or penetration testers by:

- Locating hidden assets or endpoints where vulnerabilities are more likely
- Streamlining target discovery for efficient vulnerability exploitation

Why it's important: Target discovery maximizes the chances of identifying security issues in a competitive bug bounty environment or during limited-time assessments.

2.9 Promote Security Awareness

Objective: Help the target organization understand the extent of their exposed assets and associated risks by:

- Demonstrating how attackers might view their infrastructure
- Providing insights into overlooked vulnerabilities

Why it's important: Educating stakeholders leads to proactive security measures and a more robust defense strategy.

2.10 Adapt and Customize Tools

Objective: Continuously improve the reconnaissance process by:

- Adding custom tools or modifying existing ones (e.g., Recon.sh workflows)
- Adapting methodologies to suit unique project requirements

Why it's important: Tailored tools and approaches increase the precision and relevance of the findings.

CHAPTER 3

TOOLS AND DEPENDENCIES

1.1 Installation Guide for Recon.sh

To use **Recon.sh**, you need to ensure that the required tools and dependencies are installed on your system. This guide provides step-by-step instructions to set up the script and its dependencies.

Step 1: Install Recon.sh

1. Clone the Repository:

Download the script from its repository (if hosted on GitHub or a similar platform).

```
git clone https://github.com/username/recon.sh.git
cd recon.sh
```

Make the Script Executable:

Ensure the script has executable permissions.

```
chmod +x recon.sh
```

Step 2: Install Required Tools and Dependencies

Recon.sh relies on several tools for its functionality. Below are the tools you need and their installation commands.

1. Subfinder

Subfinder is used for subdomain enumeration.

- **Installation**:

```
go install -v github.com/projectdiscovery/subfinder/v2/cmd/subfinder@latest
```

2. **Assetfinder**
   Assetfinder discovers subdomains using various APIs.

- **Installation**

```
go install github.com/tomnomnom/assetfinder@latest
```

3. **crt.sh** (via cURL and jq)
   `crt.sh retrieves SSL certificate transparency logs.

- **Install cURL**:

```
sudo apt install curl
```

- **Install jq:**

```
sudo apt install jq
```

HTTPx

HTTPx is used to detect live URLs and perform additional checks.

- **Installation**:

```
go install -v github.com/projectdiscovery/httpx/cmd/httpx@latest
```

Waybackurls

Waybackurls retrieves historical URLs for a given domain.

- **Installation**:

```
go install github.com/tomnomnom/waybackurls@latest
```

Step 3: Set Up Go Environment

Some tools require Go (Golang) to be installed.

- **Install Go**

```
sudo apt install golang-go
```

- **Verify Installation:**

```
go version
```

Step 4: Optional Tools (Enhancements)

Amass (for extended subdomain enumeration):

- **Installation**:

```
sudo apt install amass
```

- **Verify Installation**:

```
amass -h
```

Step 5: Test the Setup

1. Run the script with a sample domain to verify that all tools are working:

```
./recon.sh example.com
```

Check the output directories (e.g., example.com/subdomains/, example.com/scans/) for results.

Troubleshooting Installation Issues

1. **Command Not Found**:
   - Ensure the tool is installed and added to the system PATH.
   - Check the installation directory for binaries (e.g., $HOME/go/bin).
2. **Permission Denied**:
   - **Use chmod +x to make the script or tool executable.**
3. **Dependency Issues**:
   - **Ensure all dependencies are installed using package managers like apt, pip, or go.**

Introduction to Subfinder, Assetfinder, crt.sh, HTTPx, and Others

4.2 Subfinder

Purpose:
Subfinder is a fast and reliable subdomain discovery tool that queries various data sources, including APIs and passive DNS, to identify subdomains associated with a domain.

Key Features:

- Retrieves subdomains from multiple sources such as VirusTotal, Shodan, and ThreatCrowd.
- Supports recursive subdomain discovery.
- Outputs results in a clean and organized format.

Example Usage:
subfinder -silent -d example.com

Output:
text sub1.example.com sub2.example.com mail.example.com

4.3 Assetfinder

Purpose:
Assetfinder is a lightweight subdomain discovery tool that focuses on passive data collection from APIs.

Key Features:

- Works well for discovering subdomains without actively querying the target.
- Complements tools like Subfinder to ensure comprehensive coverage.

Example Usage:
assetfinder example.com

Output:
subdomain.example.com api.example.com admin.example.com

4.4 crt.sh

Purpose:
crt.sh is a certificate transparency log search engine that retrieves subdomains associated with SSL certificates.

Key Features:

- Finds subdomains listed in historical and active SSL certificates.
- Useful for uncovering subdomains that other tools might miss.

Example Usage in Recon.sh:
curl -s https://crt.sh//?q\=%example.com\&output\=json | jq -r '.[].name_value'

Output:
example.com sub.example.com mail.example.com

4.5 HTTPx

Purpose:
HTTPx is a versatile HTTP toolkit designed to detect live URLs, check for HTTP/HTTPS support, and retrieve server information.

Key Features:

- Verifies which subdomains are active by sending HTTP requests.

- Supports advanced features like custom headers, TLS certificate inspection, and response filtering.
- Can be used for screenshot automation when paired with system Chrome.

Example Usage: **httpx -l subdomains.txt**

Output:
https://sub1.example.com

http://sub2.example.com

4.6 GAU (Get All URLs)

Purpose:
GAU extracts URLs from various sources, including the Wayback Machine, Common Crawl, and VirusTotal.

Key Features:
- Helps uncover historical and active URLs.
- Useful for identifying endpoints and testing legacy systems.

Example Usage:
echo example.com | gau

Output:
https://example.com/api/v1/users https://example.com/login

4.7 Waybackurls

Purpose:
Waybackurls retrieves URLs archived by the Wayback Machine for a given domain.

Key Features:
- Helps uncover endpoints that may no longer be publicly accessible but could still have vulnerabilities.
- Complements GAU for historical URL extraction.

Example Usage:
echo example.com | waybackurls

Output:
https://example.com/ol
dpage
https://example.com/d
eprecated

4.8 Nmap

Purpose:

Nmap is a network scanning tool that detects open ports, services, and potential vulnerabilities on target systems.

Key Features:

- Supports advanced scans, including service detection and OS fingerprinting.
- Identifies exposed ports and services that could be entry points for attackers.

Example Usage:

nmap -p 80,443,22 example.com

Output:

PORT STATE SERVICE
22/tcp open ssh 80/tcp open http 443/tcp open https

CHAPTER 4

STEP-BY-STEP EXPLANATION OF RECON.SH

Recon.sh is a powerful tool designed for automating reconnaissance activities. Here's a step-by-step explanation of the various features it provides:

5.1 Subdomain Enumeration:

Subdomain enumeration is the process of discovering subdomains associated with a target domain. These subdomains can provide valuable insights into the infrastructure of the target. Recon.sh typically uses a combination of passive and active methods to gather this information.

How it works:

- **Passive methods:** These involve searching for publicly available information, such as DNS records, that may reveal subdomains. Tools like Sublist3r or Amass might be used.
- **Active methods:** These involve querying DNS servers directly for subdomain information. This can be done using tools like dnsdumpster or brute-forcing subdomains with a predefined wordlist.

5.1 Live Subdomain Detection:

After discovering subdomains, Recon.sh checks whether these subdomains are live (i.e., whether they resolve to an active IP address).

How it works:

- The discovered subdomains are pinged or queried to verify if they are live.
- Tools like httpx or subfinder may be used to send HTTP requests to the subdomains to check for responsiveness.

How it works:

GAU (GetAllUrls): This tool queries multiple sources like Google, Bing, and others to extract URLs associated with a domain.
Waybackurls: This tool queries the Wayback Machine to find old URLs that were once accessible but may still be useful.

5.5 Port Scanning with Nmap:

Port scanning is a key step in identifying open ports and services running on a target domain or IP address. Recon.sh automates port scanning using Nmap, which is one of the most popular and effective tools for this purpose.

How it works:

Nmap scans the target IPs for open ports and detects services running on those ports (such as HTTP, FTP, SSH).
Recon.sh automates the process to scan multiple subdomains or IP addresses at once, identifying potential entry points for an attacker.

5.6 Parameter Discovery with Paramspider:

Paramspider is a tool that helps in discovering URL parameters that may be vulnerable to attacks like SQL injection, XSS, or others. It identifies parameters by crawling the target's web pages and extracting URL parameters that can be further tested for security weaknesses.

How it works:
Paramspider crawls through URLs and extracts parameters like id=, user=, token=, etc.
These parameters can then be used for manual or automated testing for vulnerabilities (e.g., by using tools like Burp Suite or OWASP ZAP).
By automating these processes, Recon.sh enables security professionals or ethical hackers to perform comprehensive reconnaissance on a target system or domain with minimal manual effort. It covers a wide range of activities, from identifying subdomains to testing for potential vulnerabilities, making it an invaluable tool in any security assessment or penetration testing toolkit.

How the Results Are Used

1. **Bug Bounty Hunting:** Helps hunters identify vulnerabilities in exposed systems to report and earn rewards.
2. **Penetration Testing:** Guides testers to focus on high-priority areas for deeper analysis.
3. **Security Posture Improvement:** Enables organizations to identify and fix security gaps before attackers exploit them. By systematically gathering, analyzing, and reporting data, reconnaissance ensures a strong foundation for identifying and addressing vulnerabilities in any security engagement.

CHAPTER 10

FINAL RESULT AND OUTPUT

Final Result and Output of Reconnaissance

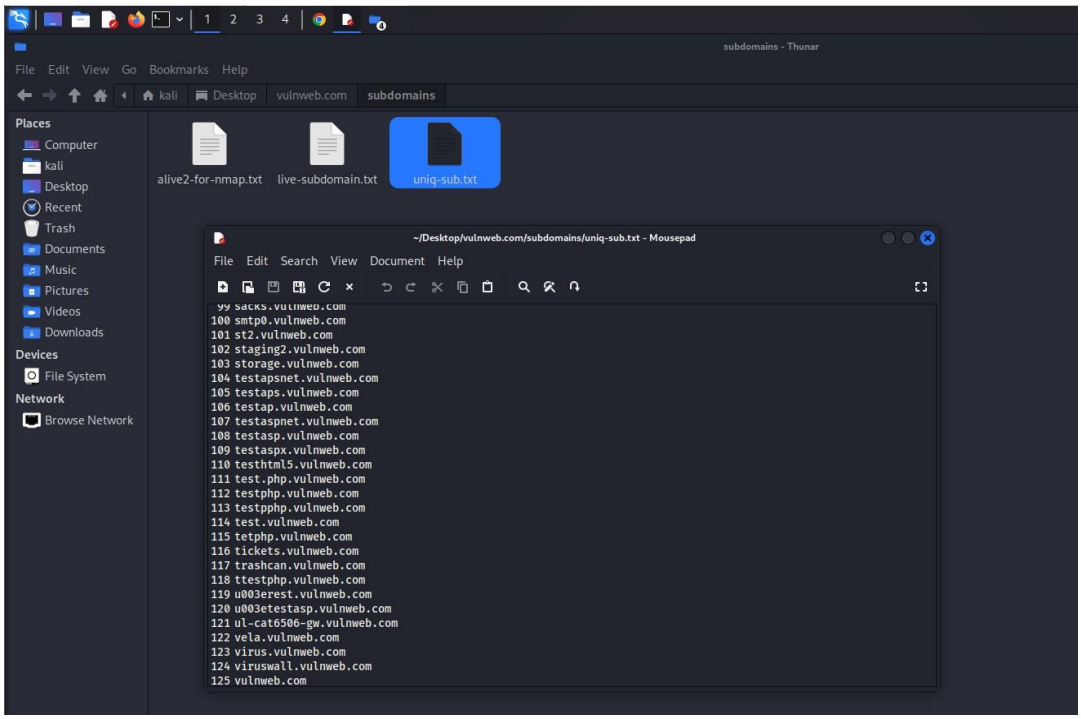
The ultimate outcome of reconnaissance is a comprehensive understanding of the target's attack surface, potential vulnerabilities, and recommendations for improving security. Below is a detailed breakdown of the **final result and outputs** generated during the reconnaissance phase.

1. Final Results

The final results of reconnaissance include tangible data and actionable insights:

Identified Assets

- **Subdomains:** A list of all discovered subdomains of the target domain.
  - Example: www.example.com, api.example.com, dev.example.com.



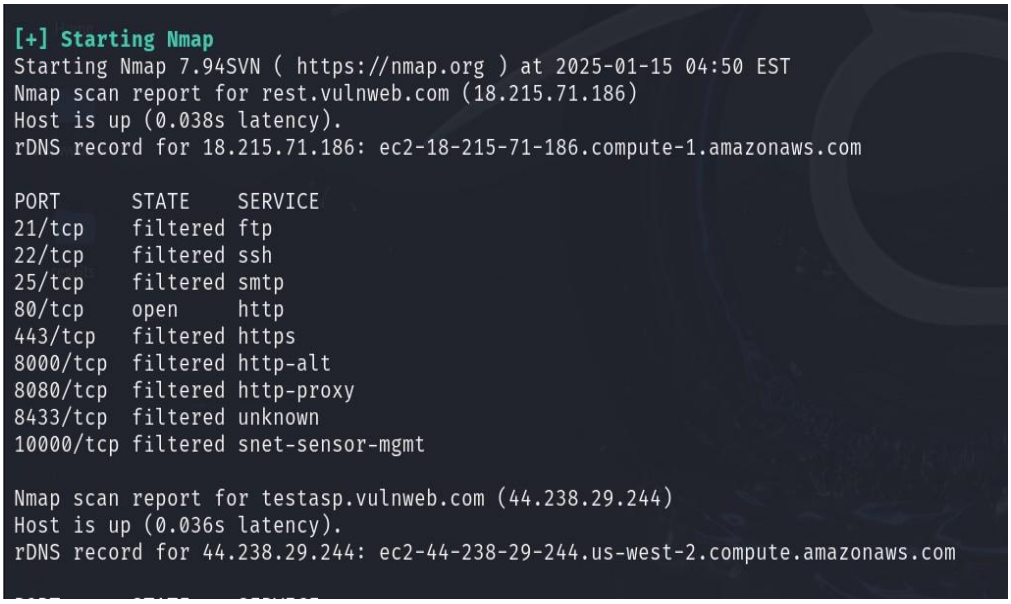
Example: 192.168.1.1, 203.0.113.5.

- **Associated Domains:** Other domains or subdomains linked to the same organization.

b. Open Ports and Service.

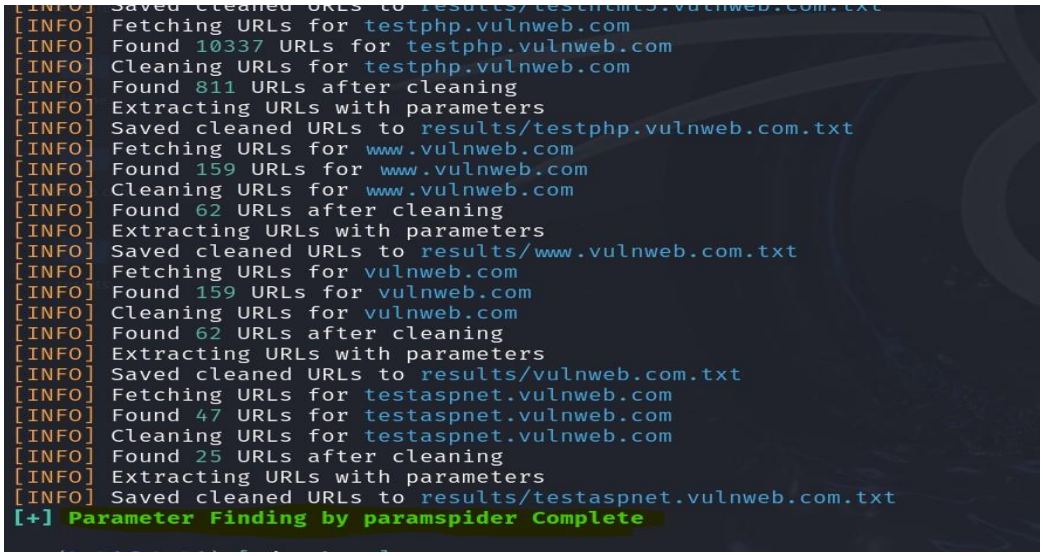
List of open ports and the services running on them.

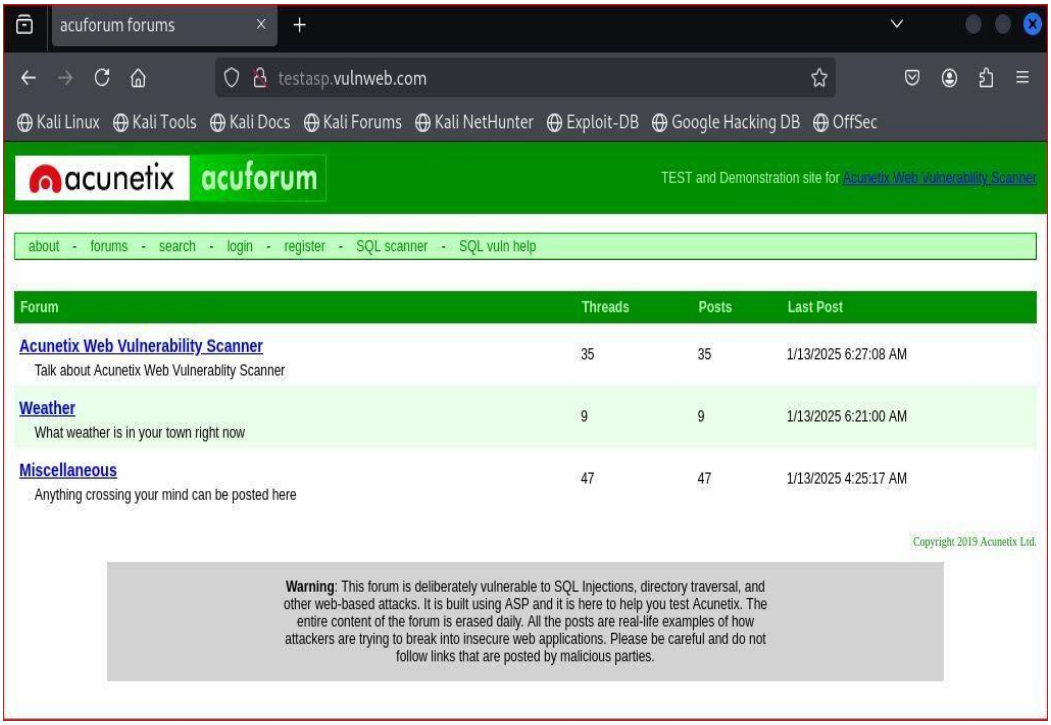
- Example:
  - Port 80 (HTTP)
  - Port 443 (HTTPS)
  - Port 22 (SSH)
- Service versions and configurations identified (e.g., **Nginx/1.18.0**, **OpenSSH 8.4**



c. *Exposed Data*

- Sensitive files, directories, or APIs discovered during scans.
  - Example:
    - /backup.zip found via directory brute-forcing.
    - API endpoints such as /api/v1/users.





More information:

Technologies Used

Fingerprinting of technologies, frameworks, and software.
Example:
CMS: WordPress 5.8.1
Web Framework: Django 3.2
Database: MySQL 8.0

d. Vulnerabilities Discovered

- Misconfigurations, outdated software, or potential vulnerabilities identified.
  - Example:
    - **Vulnerability:** Outdated Apache Server (2.4.29) with CVE-2019-0211.
    - **Misconfiguration:** Missing HTTP security headers like Content-Security-Policy.

e. Risk Assessment

- Categorization of risks based on the likelihood of exploitation and potential impact.
  - **High-Risk:** Exposed database with default credentials.
  - **Medium-Risk:** Outdated SSL/TLS version.
  - **Low-Risk:** Missing HTTP headers.

f. Visual Outputs (Optional)

- **Screenshots:** Images of live subdomains for easier analysis of web interfaces.
- **Network Maps:** Visual representation of open ports and connections (e.g., Nmap topology graphs).

2. Output Files and Data Formats

Reconnaissance outputs are typically structured into organized files and folders for analysis and reporting.

File Types:

- **Subdomains:** subdomains.txt
- **Live Hosts:** live_hosts.txt
- **Open Ports:** ports.txt
- **Technology Details:** technologies.csv
- **Vulnerabilities:** vulnerabilities.txt
- **API Endpoints:** api_endpoints.txt

Sample Directory Structure:

```
/recon-results/
├── subdomains.txt
├── live_hosts.txt
├── open_ports.txt
├── technologies.csv
├── vulnerabilities.txt
├── api_endpoints.txt
├── screenshots/
│   ├── www.example.com.png
│   └── api.example.com.png
└── nmap_scan.xml
```

3. Actionable Insights

The output includes actionable recommendations based on the findings:

Example Recommendations:

1. **Fix High-Risk Vulnerabilities:**
   - Update Apache server to the latest version to patch CVE-2019-0211.
   - Restrict access to /backup.zip or remove the file entirely.
2. **Improve Security Configurations:**
   - Enforce HTTP security headers (e.g., Strict-Transport-Security, Content-Security-Policy).
   - Disable unused ports (e.g., close Port 21 if FTP is not in use).
3. **General Improvements:**
   - Implement rate-limiting on API endpoints to prevent abuse.
   - Regularly scan for exposed assets and vulnerabilities.

4. Final Deliverable

The final deliverable is a **reconnaissance report** that provides stakeholders with a clear understanding of their attack surface and security posture. The report should include:

1. **Executive Summary:** High-level overview of findings.
2. **Detailed Findings:** Comprehensive list of identified assets, vulnerabilities, and risks.
3. **Evidence:** Screenshots, logs, and tool outputs supporting the findings.
4. **Recommendations:** Specific steps to address vulnerabilities and improve security.

CHAPTER 12

CONCLUSION AND FUTURE WORK

Limitations

- **Dependency on External Tools**: The tool relies on several third-party utilities, which may not always be available or up-to-date.
- **False Positives**: Automated tools may sometimes yield incorrect results that require manual verification.
- **Network Constraints**: Performance is affected by the quality of the network and the target's server configurations.
- **Learning Curve**: Users need a basic understanding of Bash scripting and the integrated tools to fully utilize the system.

Future Work

- **Enhanced Compatibility**: Extend support for more tools and technologies to improve coverage and accuracy.
- **AI Integration**: Incorporate machine learning to reduce false positives and enhance result validation.
- **GUI Development**: Create a user-friendly graphical interface to simplify usage for non-technical users.
- **Real-Time Reporting**: Enable real-time updates and analytics for ongoing assessments.
- **Cloud Integration**: Allow deployment on cloud platforms for scalability and accessibility.

Conclusion

The "Automation Tools for Web Security" project successfully addresses the challenges of manual reconnaissance by integrating multiple tools into a cohesive and efficient Bash script. While the tool has limitations, its benefits in time-saving, comprehensiveness, and ease of use make it a valuable asset for cybersecurity professionals. Future enhancements, including AI integration and GUI development, hold promise for making the tool even more versatile and accessible.

Reference

Below are reference links for the tools and commands used in the script, categorized for each functionality. These references provide official documentation or GitHub repositories to help understand their usage in detail.

1. Subdomain Enumeration

Subfinder
Description: Subdomain discovery tool that uses passive sources.
GitHub Link: https://github.com/projectdiscovery/subfinder
Assetfinder
Description: A subdomain discovery tool that finds related domains.
GitHub Link: https://github.com/tomnomnom/assetfinder
crt.sh

Description: A public database of Certificate Transparency logs used for subdomain discovery.
Website: https://crt.sh/
Usage Example (curl + jq): jq Documentation

2. Live Subdomain Detection

HTTPx

Description: Fast and multi-purpose HTTP toolkit for detecting live subdomains and additional information.
GitHub Link: https://github.com/projectdiscovery/httpx

3. Screenshot Automation

Gowitness
Description: A tool for taking screenshots of web pages using headless Chrome.
GitHub Link: https://github.com/sensepost/gowitness

4. URL Extraction

GAU (GetAllUrls)

Description: Fetches URLs from various sources such as Wayback Machine, Common Crawl, and VirusTotal.
GitHub Link: https://github.com/lc/gau
Waybackurls
Description: Retrieves URLs from the Wayback Machine and other archive services.
GitHub Link: https://github.com/tomnomnom/waybackurls

5. Port Scanning

Nmap

Description: A network discovery and security auditing tool used to perform port scanning and service detection.
Official Website: https://nmap.org/ -
Documentation: https://nmap.org/docs.html

6. Parameter Discovery

ParamSpider
Description: A tool to discover GET parameters for a given domain.
GitHub Link: https://github.com/devanshbatham/ParamSpider

7. Script Enhancements and Debugging

Bash Scripting
Description: Learn about advanced Bash scripting techniques to optimize and debug scripts.
Bash Reference Manual: https://www.gnu.org/software/bash/manual/
Color Formatting in Bash: https://tldp.org/LDP/abs/html/colorizing.html

8. Ethical Use of Recon Tools

OWASP Testing Guide: https://owasp.org/www-project-web-security-testing-guide/
Legal and Ethical Guidelines: https://hackerone.com/disclosure-guidelines These references will help you understand and verify each step of the script. Let me know if you need further clarification!