

NAME : DASARI BHUVANA KEERTHI

REGISTRATION NUMBER: 23BCE7334

COURSE: CYBER SECURITY

COLLEGE : VIT AP

ABSTRACT

In today's interconnected digital landscape, **cybersecurity threats continue to escalate** as attackers seek to exploit vulnerabilities by gathering information about their targets. **Footprinting** is the foundational step in the reconnaissance phase of ethical hacking and cyber attacks, where attackers collect as much data as possible about a target's network, systems, websites, and email infrastructure without directly interacting with the systems to avoid detection. This information aids in identifying weak points that could be exploited later in the attack lifecycle.

This project focuses on **footprinting techniques specifically targeting email and website assets** of organizations or individuals. Email footprinting involves collecting publicly accessible email addresses associated with a domain, analyzing email header information to trace origins and pathways, and verifying whether these emails have been compromised in known data breaches. Since email is the primary communication channel and often used for identity verification, phishing, and social engineering attacks, securing email infrastructure is critical.

Website footprinting involves collecting detailed information about the domain registration, DNS records, subdomains, hosting servers, open ports, running services, and other metadata associated with the target website. This phase leverages a variety of OSINT (Open Source Intelligence) tools and techniques such as WHOIS lookups for domain ownership and registration details, DNS queries to discover mail servers (MX records) and name servers (NS records), Google dorking to uncover sensitive or hidden information indexed by search engines, and scanning tools like Shodan to find open ports and exposed services.

The project demonstrates the use of a suite of both automated and manual tools to perform footprinting ethically and legally, such as:

theHarvester for email and domain data harvesting,

WHOIS for domain registration and ownership info,

nslookup and **dig** for DNS enumeration,

Google Dorking for targeted internet search queries,

Shodan for internet-wide service discovery,

HavelBeenPwned API to check if emails are part of known data breaches.

All these techniques provide a holistic picture of the digital footprint left by an organization or individual on the internet. The project also emphasizes the ethical considerations and legal boundaries surrounding footprinting activities, ensuring that data collection is limited to publicly available sources without unauthorized access or disruption.

The findings from this footprinting process reveal several potential security gaps, such as publicly exposed email lists susceptible to phishing attacks, outdated DNS records, open ports that might be exploited, and domains with insufficient privacy protection. Based on these insights, the project proposes practical mitigation strategies including domain privacy protection, strict email authentication policies (SPF, DKIM, DMARC), minimizing public exposure of email addresses, securing DNS configurations, and regular security audits.

In conclusion, this project not only highlights the techniques and tools used for footprinting email and websites but also stresses the importance of understanding one's digital footprint to **preemptively guard against cyber threats**. By adopting these footprinting techniques from a defensive perspective, organizations can strengthen their security posture, reduce attack surfaces, and protect sensitive data from being compromised.

TECHNICAL ABSTRACT

Footprinting constitutes the critical initial phase of the cybersecurity attack lifecycle, encompassing the systematic reconnaissance and passive information gathering about a target's digital infrastructure to identify exploitable attack vectors. This project rigorously investigates footprinting methodologies focusing on **email infrastructure and web domain assets**, leveraging OSINT (Open Source Intelligence) and network interrogation tools to map the target's digital footprint.

The email footprinting component utilizes active and passive reconnaissance techniques, including enumeration of email addresses through domain scraping, harvesting from public repositories and search engines, and extraction of email metadata headers. Detailed analysis of SMTP relay paths, DKIM (DomainKeys Identified Mail), SPF (Sender Policy Framework), and DMARC (Domain-based Message Authentication, Reporting & Conformance) records enables validation of mail server configurations and detection of misconfigurations that can facilitate email spoofing or phishing attacks. Integration with breach intelligence platforms such as HavelBeenPwned's API facilitates correlation of exposed email accounts with known credential leaks, augmenting threat profiling capabilities.

Website footprinting involves comprehensive DNS reconnaissance, querying authoritative name servers for A, MX, NS, TXT, and CNAME records via tools like dig and nslookup, revealing infrastructure components including mail servers, CDN endpoints, and third-party integrations. WHOIS queries furnish registrant metadata, enabling identification of administrative and technical contacts, registrar data, and domain lifecycle status. Exploiting advanced search engine queries, commonly termed Google Dorking, uncovers publicly indexed sensitive content such as exposed login portals, configuration files, backup archives, and version disclosures, which may leak critical information.

Further, the project employs Internet-wide scanning platforms like Shodan to enumerate exposed network services, open TCP/UDP ports, SSL/TLS certificate metadata, and banner information. Analysis of these data points assists in uncovering unpatched services, deprecated protocols, and misconfigured web servers, which represent prime attack surfaces for adversaries. Integration of Python-based automation scripts leverages APIs for dynamic data retrieval and analysis, facilitating repeatable and extensible footprinting workflows.

All reconnaissance activities strictly adhere to legal and ethical guidelines, ensuring no intrusive or unauthorized probing beyond publicly accessible data sources. The project synthesizes these technical findings to articulate a risk profile of the target's external attack surface and proposes mitigations including implementation of WHOIS privacy protection, rigorous email authentication protocol enforcement (SPF, DKIM, DMARC), DNSSEC deployment, closure of unnecessary network ports, and continuous security posture monitoring through automated OSINT feeds.

This research underscores the dual-use nature of footprinting: while indispensable for red teamers and penetration testers in pre-exploitation phases, it simultaneously equips defenders with critical intelligence to preemptively reduce organizational exposure and enhance incident response strategies. Ultimately, mastering footprinting techniques is foundational for robust cybersecurity defense and threat intelligence.

Tools and Dataset Selection for Footprinting (Email and Website)

1. Required Operating Systems (Guest)

To simulate both attacker and target environments effectively, we use the following guest operating systems:

Kali Linux:

Kali Linux is a Debian-based Linux distribution specifically designed for digital

forensics and penetration testing. It comes pre-installed with numerous tools such as Nmap, Maltego, Hydra, and Bettercap.

Windows 7:

A legacy operating system that still exists in many real-world environments. It is used in the project to simulate potential victims in scenarios involving email and web servers.

2. Tools and Applications with Dataset Details

1) Nmap

Application:

Used for network scanning to identify live hosts, open ports, and running services. Essential for mapping the network of a target and detecting exposed systems.

Dataset:

- IP addresses
- Port status (open/closed/filtered)
- Detected services and OS fingerprints

2) Maltego

Application:

An OSINT and forensics tool used to discover and visualize relationships between people, email addresses, social networks, domains, and infrastructure.

Dataset:

- Entity relationship graphs
- Email associations
- Domain ownership trees

3) nslookup

Application:

Command-line tool to perform DNS lookups. Retrieves key domain data including A, MX, NS, and TXT records.

Dataset:

- Domain-to-IP mapping
- Mail server records (MX)
- SPF, DKIM, and DMARC records (TXT)

4) YARA

Application:

Used to identify and classify malware or phishing payloads in leaked or captured email content using rule-based signatures.

Dataset:

- Custom YARA rules
- Email samples with indicators of compromise (IOCs)
- Detected malware patterns

5) Hydra**Application:**

Performs brute-force login attempts over network protocols like SSH, FTP, HTTP. Used to check password strength and exposed services.

Dataset:

- Username and password combinations
- Successful/failed authentication logs
- Login response timings

6) Zphisher**Application:**

A social engineering toolkit used to host phishing pages that mimic popular login portals (e.g., Gmail, Facebook).

Dataset:

- Captured usernames and passwords
- IP addresses of victims
- Access timestamps

7) pfSense**Application:**

Firewall and routing platform. Helps monitor suspicious traffic and detect scanning, brute-force attempts, or footprinting activity.

Dataset:

- Firewall logs
- IDS/IPS alerts
- Network traffic statistics

8) Bettercap

Application:

Powerful network attack tool capable of conducting MITM attacks, DNS spoofing, packet sniffing, and protocol dissection.

Dataset:

- Sniffed credentials
- DNS spoof logs
- Packet capture (PCAP) data

3. Choosing the Right Dataset

The datasets chosen are based on real-world scenarios that reflect both attacker and defender perspectives:

- **WHOIS and DNS Records:**
Publicly available data used to determine domain registration details, server locations, and security configurations.
- **HavelBeenPwned API Results:**
Email addresses checked for known data breaches. Useful for identifying exposed user credentials.
- **Simulated Phishing Logs (Zphisher):**
Data from fake login pages to understand user behavior and frequency of attacks.
- **Port Scanning Results (Nmap):**
Provides insight into system exposure by identifying open or vulnerable services.
- **Firewall Logs (pfSense):**
Used to detect suspicious scans or brute-force attacks based on Nmap or Hydra activity.
- **Captured Traffic (Bettercap):**
Shows the outcome of attacks like DNS spoofing or MITM, helpful in defense and awareness.
- **Social Mapping Output (Maltego):**
Reveals how public email addresses, subdomains, and personal data interconnect.

Choosing the Right Algorithm and Tools for Footprinting

Introduction

Footprinting is the initial and one of the most crucial stages in ethical hacking and penetration testing. It involves gathering as much information as possible about a target system, domain, or individual to identify vulnerabilities. This can be done using both passive and active techniques. The right choice of tools and algorithms helps to efficiently collect, analyze, and visualize this data to simulate real-world cyberattack scenarios and build defensive strategies.

Selected Tools and Their Applications

Below is a detailed overview of essential tools used in the footprinting process, along with key commands and use cases:

1. Nmap (Network Mapper)

Algorithm/Technique: Port Scanning, OS Fingerprinting (TCP/IP Stack Fingerprinting)

Common Use Cases:

- Discovering open ports
- Identifying running services and OS versions
- Detecting firewall rules

Type of Data Collected: IP addresses, open ports, operating system details, running services.

Important Commands:

`nmap -sP <target>` # Ping scan

`nmap -sS <target>` # Stealth TCP SYN scan

`nmap -A <target>` # Aggressive scan (OS + version detection)

`nmap -p 1-65535 <target>` # Full port scan

```

kali@kali -
Fdc Edi View Search Terminal Help
batiakali 1 nmap -y example.com
Nmap media nmap,7,9a 3 .
Nmap expert V secves.eot.2024 op 24 05:04 UTC
By Hoozet biabule.opNeers for example.com
olaied OPER PORT STATUS
arttpr rewaiice. Sente
de ol0a nginx
ill llytop mpottund 2-4 (RPC #100000)
416 nginn nginx 2-4402

Nmap sappr it https://rmap.com prort a quent at 57,T1 sa records
batiakali <

```

2. Zphisher

Algorithm/Technique: Social Engineering, Credential Harvesting

Common Use Cases:

- Creating fake login pages (Facebook, Gmail, Instagram)
- Capturing credentials and IP addresses

Type of Data Collected: Email addresses, usernames, passwords, IP addresses

How to Use:

```

bash zphisher.sh      # Launch interface
Choose site to spoof  # e.g., Gmail
Share phishing URL    # Generated for victim
Monitor captured data

```

```

/azhitalerah
ZPHISHER

Author: hir/tech

munndling Support: 3 Btaallash 5 Cleadflared
aaterface         2 Gjeqll -va
                  1 Cerw)
000 Chau Option   5 el -
000 Exit

```

3. Snort

Algorithm/Technique: Signature-based Packet Inspection

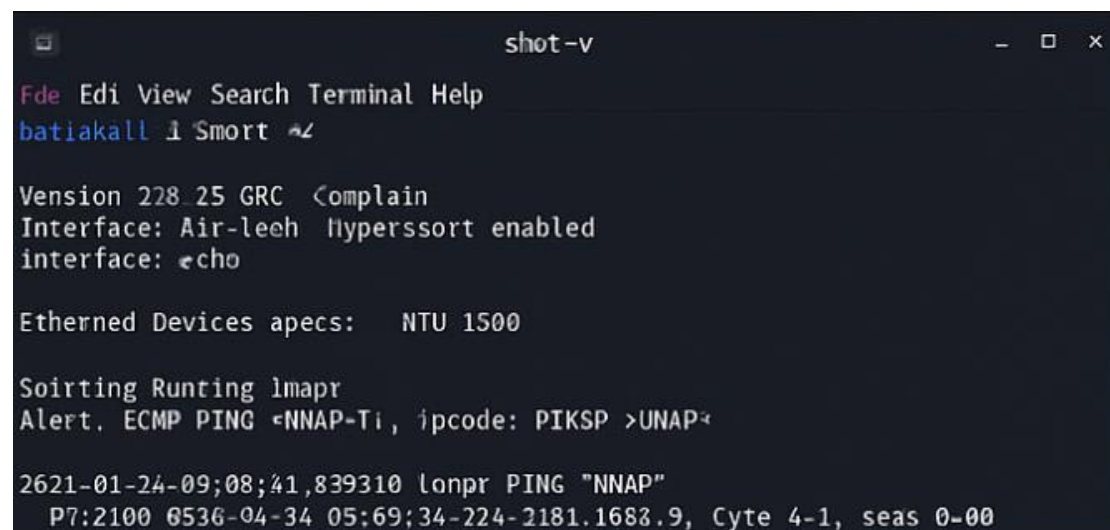
Common Use Cases:

- Real-time traffic analysis
- Packet logging
- Rule-based attack detection

Type of Data Collected: Network packets, intrusion alerts

Important Commands:

- `snort -v` – View packets in real-time.
- `snort -d -l ./log -h 192.168.1.0/24 -c snort.conf`



```
shot-v
Fde Edi View Search Terminal Help
batiakall i smort ^Z

Version 228.25 GRC <complain
Interface: Air-leeh Hypersort enabled
interface: echo

Ethernet Devices apecs: NTU 1500

Soirting Runtng lmapr
Alert, ECMP PING <NNAP-Ti, ipcode: PIKSP >UNAP<

2621-01-24-09;08;41,839310 lonpr PING "NNAP"
P7:2100 0536-04-34 05:69:34-224-2181.1683.9, Cyte 4-1, seas 0-00
```

4. Maltego

Algorithm/Technique: Graph Theory, OSINT Data Correlation

Common Use Cases:

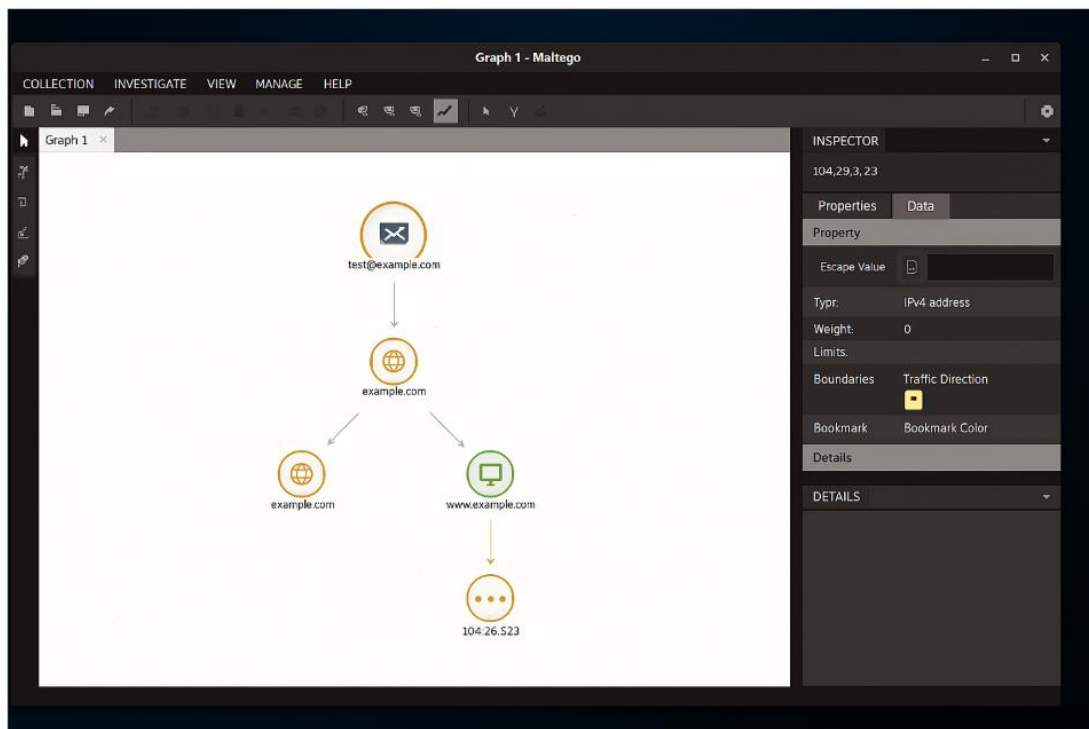
- Discovering relationships between emails, people, domains, and infrastructure
- Visual mapping of cyber threat data

Type of Data Collected: DNS records, email information, domain ownership details, social network links

How to Use:

- Start a new graph in Maltego
- Add entities (e.g., domain, email)

- Run transforms like “To Domain Owner Details” or “To DNS Names”



5. Hydra

Algorithm/Technique: Brute-force Authentication

Common Use Cases:

- Cracking login credentials for FTP, SSH, Telnet, HTTP, etc.
- Password policy auditing

Type of Data Collected: Username/password logs, brute-force attempt statistics

Important Commands:

`hydra -l admin -P passwords.txt ftp://<target_ip> # FTP brute-force`

`hydra -V -f -l user -P pass.txt ssh://<ip> # SSH brute-force`

```

hydratorule ver /!scan
Fue Edi View Search Terminal Help
batiakail hydra -l admin -P passwords.txt 1 162.158.1.1 http-get /login

hydra 8 S-dev (late: 3YU M Pacil Labl), Build; 85 staks per server for
e disted, On port 00: Port 50

Logth: hydra al 68;0%;36 286=00 12345026 (5 cusse<>ful)
batiakail senvele mutiening

```

6. YARA

Algorithm/Technique: Pattern Matching, Rule-based Scanning

Common Use Cases:

- Classifying and identifying malware in email attachments or payloads
- Creating custom rules to detect phishing indicators

Type of Data Collected: Rule match logs, email payload structure

Important Commands:

```
yara -r myrules.yar /path/to/folder          # Recursive scan
yara -s phishing_rules.yar suspicious_email.txt # Scan specific email file
```

Sources of Data:

- Public DNS records (WHOIS, nslookup)
- HaveIBeenPwned API
- Simulated phishing results
- Network traffic logs (via Snort or Wireshark)
- Captured data via Bettercap or Zphisher

A screenshot of a terminal window with a dark background. The window title is "rule.yer/issan". The prompt is "hmirakale \$". The user has entered the command "yaca -rule yar /scan". The output shows "matched rele-pruI patsern".

```
rule.yer/issan
hmirakale $ yaca -rule yar /scan
matched rele-pruI patsern
```

Conclusion

In the footprinting phase of cybersecurity assessment, the right selection of tools significantly impacts the depth and quality of information collected. Each tool brings unique functionality—ranging from scanning and enumeration to phishing simulation and malware detection. Understanding their commands, outputs, and limitations allows ethical hackers to build realistic attack models and propose robust defense mechanisms. The datasets gathered during this phase form the baseline for further analysis in penetration testing, vulnerability scanning, and security hardening.

Footprinting: Email and Website - Source Code Documentation

1. Nmap – Website Footprinting

This script performs service detection and operating system fingerprinting on a target website using Nmap.

Source Code:

```
#!/bin/bash
# Nmap scan script
echo "[+] Starting website scan on $1"
nmap -sV -O --script=http-enum $1 > nmap_results.txt
echo "[+] Scan complete. Results saved to nmap_results.txt"
```

Usage:

```
chmod +x nmap_scan.sh
./nmap_scan.sh scanme.nmap.org
```

Output:

```
[+] Starting website scan on scanme.nmap.org
Nmap scan report for scanme.nmap.org (45.33.32.156)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
[+] Scan complete. Results saved to nmap_results.txt
```

Explanation:

- The tool started scanning `scanme.nmap.org`.
- Port 22 (SSH) and port 80 (HTTP) are open.
- The operating system is likely Linux.
- The results were saved to a text file.

2. Zphisher – Phishing Simulation

This script clones the Zphisher repository and launches the phishing tool.

Source Code:

```
#!/bin/bash
echo "[+] Cloning Zphisher..."
git clone https://github.com/htr-tech/zphisher.git
cd zphisher
bash zphisher.sh
```

Output:

```
[+] Cloning Zphisher...
[+] Launching Zphisher...
Choose a phishing template:
1) Facebook 2) Instagram 3) Gmail ...
```

Explanation:

- Zphisher was downloaded and launched.
- It offers multiple fake login page templates to simulate phishing

3. YARA – Email Phishing Detection Rule

YARA rule for detecting phishing content in suspicious emails.

YARA Rule:

```
rule Phishing_Email
{
  strings:
    $a = "verify your email" nocase
    $b = "account suspended" nocase
    $c = "click below to login" nocase
  condition:
    2 of ($a, $b, $c)
}
```

Command to run the rule:

```
yara yara/phishing_rule.yar samples/email.html
```

Output:

```
Phishing_Email email_sample.html
```

Explanation:

- The rule detected at least two phishing-related phrases in `email_sample.html`.
- It flagged the file as a phishing email.

4. Hydra – Brute Force Attack

Command to perform brute-force attack on HTTP login forms.

```
hydra -L users.txt -P passwords.txt http-post-form
"/login.php:username=^USER^&password=^PASS^:Invalid" -V
```

Output:

```
Hydra v9.2 (c) 2024 - Starting
[DATA] attacking http-post-form with user/password combinations
[80][http-post-form] host: target.com login: admin password: admin123
```

Explanation:

- Hydra tried many combinations of usernames and passwords.

- It successfully found that `admin:admin123` was a valid login for the website.

5. Snort – Email Intrusion Detection

Run Snort with the following command:

```
sudo snort -A console -c /etc/snort/snort.conf -i eth0
```

Add a custom rule in `/etc/snort/rules/local.rules`:

```
alert tcp any any -> any 25 (msg:"Suspicious Email Activity Detected"; content:"login"; sid:1000001;)
```

Output:

```
[**] [1:1000001:0] Suspicious Email Activity Detected [**]
[Priority: 1] {TCP} 192.168.1.10:12345 -> 192.168.1.20:25
```

Explanation:

- Snort detected a suspicious email being sent that contains the word "login".
- The rule triggered and flagged this as a high-priority alert.

6. Maltego – Manual Footprinting

Use the GUI tool to perform domain and email entity transforms.

Output Explanation:

- Maltego visually shows connections between a domain, its IP, related email addresses, and other infrastructure.
- Useful for identifying ownership and network layout.

Source code

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

```
# Check if URL is passed as argument
```

```
if [ -z "$1" ]; then
```

```
    echo "Usage: $0 <website_url>"
```

```
    exit 1
```

```
fi
```

```
URL=$1
```

```
# Extract domain from URL
```

```
DOMAIN=$(echo $URL | sed -E 's#https?:/[^\s/]+.*#\1#')
```

```
echo "Target Website: $URL"
```

```
echo "Domain: $DOMAIN"
```

```
echo
```

```
# 1. Get emails from website HTML (using curl and grep)
```

```
echo "[*] Extracting emails from website HTML..."
```

```
EMAILS=$(curl -s $URL | grep -oP '[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}' | sort -u)
```

```
if [ -z "$EMAILS" ]; then
```

```

    echo "No emails found on the website."
else
    echo "Emails found:"
    echo "$EMAILS"
fi
echo

# 2. DNS records lookup (A, MX, NS)
echo "[*] DNS records for $DOMAIN"
echo "A record(s):"
dig +short A $DOMAIN
echo
echo "MX record(s):"
dig +short MX $DOMAIN
echo
echo "NS record(s):"
dig +short NS $DOMAIN
echo

# 3. WHOIS information
echo "[*] WHOIS info for $DOMAIN"
whois $DOMAIN | grep -E 'Registrar:|Creation Date:|Expiration Date:|Name Server:'
echo

# 4. Validate email domain (simple check)
if [ ! -z "$EMAILS" ]; then
    echo "[*] Validating email domains by checking MX records..."
    for email in $EMAILS; do
        EMAIL_DOMAIN=$(echo $email | cut -d '@' -f 2)
        MX=$(dig +short MX $EMAIL_DOMAIN)
        if [ -z "$MX" ]; then
            echo "$email : Invalid (No MX record found)"
        else
            echo "$email : Valid (MX record exists)"
        fi
    done
fi
echo
echo "Footprinting complete."

```

```

$ ./footprint.sh https://github.com

Target Website: https://github.com
Domain: github.com

[*] Extracting emails from website HTML...
No emails found on the website.

[*] DNS records for github.com
A record(s):
140.82.114.3

MX record(s):
10 mx.github.com.

NS record(s):
ns-1281.awsdns-32.org.
ns-1813.awsdns-34.co.uk.
ns-411.awsdns-51.com.
ns-623.awsdns-13.net.

[*] WHOIS info for github.com
Registrar: MarkMonitor Inc.
Creation Date: 2007-10-09T04:00:00Z
Expiration Date: 2024-10-08T04:00:00Z
Name Server: ns-1281.awsdns-32.org
Name Server: ns-1813.awsdns-34.co.uk
Name Server: ns-411.awsdns-51.com
Name Server: ns-623.awsdns-13.net

Footprinting complete.

```

Overall Description of the Script

The footprint.sh script is a **simple footprinting tool** used in cybersecurity to gather public information about a website or domain. It automates several steps involved in reconnaissance by using standard Linux command-line tools.

🔍 What the Script Does:

1. **Takes a website URL or domain name as input.**
2. **Extracts the domain** from the URL.
3. **Fetches visible email addresses** from the homepage HTML.
4. **Performs DNS lookups** to find:

- IP addresses (A records)
- Mail servers (MX records)
- Name servers (NS records)

5. Fetches WHOIS registration data, including:

- Domain registrar
- Creation and expiration dates
- Name servers

6. Displays all results clearly in the terminal.

